

DISSERTATION

SPARK: SECURE, PRIVACY-AWARE, AND ROBUST ALGORITHMS FOR HIGH-
ACCURACY MOBILE INDOOR LOCALIZATION USING MACHINE LEARNING

Submitted by

Danish Gufran

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2025

Doctoral Committee:

Advisor: Sudeep Pasricha

Anthony Maciejewski

Anura Jayasumana

Nikhil Krishnaswamy

Copyright by Danish Gufran 2025

All Rights Reserved

ABSTRACT

SPARK: SECURE, PRIVACY-AWARE, AND ROBUST ALGORITHMS FOR HIGH-ACCURACY MOBILE INDOOR LOCALIZATION USING MACHINE LEARNING

Navigation systems are the backbone of modern mobility, essential not only for positioning people and devices but also for enabling intelligent routing, congestion management, and personalized guidance. Outdoor navigation owes much of its success to the Global Positioning System (GPS), which has redefined how we navigate outdoors. However, GPS-based systems falter within indoor environments due to their inability to penetrate walls and obstacles. With society increasingly reliant on indoor positioning or localization—whether in airports, shopping malls, hospitals, smart homes, or factories—this limitation demands innovative alternatives. As a result, the focus has shifted toward wireless technologies such as Bluetooth, ultra-wideband (UWB), and Wi-Fi, which—when coupled with indoor localization techniques like trilateration, triangulation, or fingerprinting—can serve as the building blocks of indoor localization systems. Among these, Wi-Fi-based fingerprinting offers a promising solution, leveraging existing Wi-Fi infrastructure for low-cost and accurate localization. Recent advances in machine learning (ML) further enhance the potential of these systems. Despite these developments, implementing a reliable ML assisted indoor localization system poses several critical challenges, which we aim to address in this dissertation.

The ML assisted Wi-Fi fingerprinting-based approach relies on two key phases: (1) an offline phase, where location-tagged Wi-Fi fingerprints are collected and trained on the ML model, and (2) the online phase, where new Wi-Fi fingerprints from user devices are processed through the

pre-trained ML model to predict the user's locations in real-time. This approach offers resilience against common environmental disturbances, such as multipath interference and shadowing, while also eliminating the need for a line-of-sight (LoS) between the user and Wi-Fi access points (APs)—a limitation that significantly constrains trilateration and triangulation-based methods—making it particularly robust and practical in real-world indoor environments. Yet, it comes with several shortcomings: (i) **Balancing Accuracy, Energy Efficiency, and Latency:** Mobile devices have limited CPU power, memory, and battery life, making it challenging to run large ML models. Lightweight ML models must provide quick and reliable predictions while conserving energy and maintaining device performance. (ii) **Device Heterogeneity Resilience:** Variations in Wi-Fi chipsets, antennas, and firmware noise filtering algorithms across devices cause inconsistencies in RSS measurements, even at the same location. Addressing these differences is key to building reliable models that generalize across diverse hardware. (iii) **ML Model Security:** Wi-Fi systems are vulnerable to adversarial attacks, which manipulate RSS values and compromise location estimates. Resilience to such attacks is critical to ensure secure indoor localization and enhance the reliability of the indoor localization system. (iv) **Long-Term Indoor Localization :** Indoor spaces change frequently—furniture moves, crowd densities shift, and AP power levels fluctuate. ML models must maintain reliable performance over time in the face of such environmental changes, without requiring costly or frequent re-calibration. (v) **Adaptation to Dynamic Environments:** Most ML models are static and fail to maintain accuracy as Wi-Fi RSS signals evolve due to device heterogeneity and long-term environmental changes. Enabling the model to continually adapt to these variations is essential for maintaining high localization accuracy and reliability. (vi) **Efficient Data Sourcing and Privacy Preservation:** Gathering location-tagged fingerprints is labor-intensive and time-consuming, especially with varying environmental factors.

Intelligent data sourcing strategies must minimize manual effort while safeguarding user data through privacy-preserving mechanisms. (vii) Explainability and Transparency of ML Models: Despite delivering high accuracy, ML models often act as black box systems, making it difficult to interpret their predictions. Enhancing model explainability is vital for user trust, regulatory compliance, and debugging localization errors, particularly in dynamic environments.

To address these core challenges, this dissertation introduces *SPARK*, a collection of novel ML frameworks tailored for Wi-Fi fingerprinting-based indoor localization. A fundamental requirement for any model in this domain is the ability to deliver high localization accuracy while remaining energy-efficient enough for deployment on mobile devices. All *SPARK* frameworks are designed with this constraint in mind. To address the challenge of device heterogeneity, *SPARK* introduces novel attention-based neural networks and vision transformers to extract device-invariant features from Wi-Fi RSS to enhance robustness across devices with different hardware and software configurations.

As indoor environments are constantly evolving, ensuring long-term localization stability is essential. *SPARK* introduces a novel contrastive learning-based Siamese neural network that enables long-term localization without the need for frequent manual re-calibration of the model. To alleviate the burden of manual data collection, *SPARK* introduces a novel stacked autoencoder-based data augmentation to generate synthetic fingerprints, thereby minimizing the need for labor-intensive labeling. It also introduces a novel privacy-aware crowdsourcing federated learning method to enable decentralized model training without centralized data storage, thereby preserving user privacy while improving scalability.

To ensure the model remains adaptive in ever-changing (dynamic) environments, *SPARK* introduces novel domain-incremental learning, continual learning, and graph-based learning methods. These methods allow the model to adapt to environmental changes and learn the spatial geometry of indoor layouts, allowing adaptability in dynamic and structurally complex environments. As indoor localization systems grow more distributed and intelligent, security becomes critical. *SPARK* introduces novel capsule neural networks, curriculum-based adversarial learning, and saliency-driven federated learning methods to defend against cyberattacks, ensuring robustness under adversarial conditions. Finally, *SPARK* addresses the growing need for explainability in ML systems by introducing a novel logic gate-based method for interpreting ML model decisions. This approach offers transparent decision-making, reveals how traditional ML models behave under noisy conditions, identifies key ML decision influencing features, and enables strategic diagnosis of prediction failures in real-world deployments.

Together, the *SPARK* frameworks address several of the most critical and open challenges in Wi-Fi fingerprinting-based indoor localization. This dissertation guides future research toward the development of domain-specific, scalable, secure, and explainable indoor localization systems, validated through extensive real-world evaluations and analysis.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to all those whose support, guidance, and encouragement have made this dissertation possible.

First and foremost, I extend my heartfelt appreciation to my advisor, Prof. Sudeep Pasricha, whose insightful guidance has been instrumental throughout my doctoral journey. His expertise and mentorship have empowered me to explore innovative solutions to the challenges of modern indoor localization. Prof. Pasricha's invaluable advice, both technical and personal, has guided me through complex research problems and enhanced my attention to detail. His commitment to my academic and personal development has been unwavering, pushing me to grow not just as a researcher but also as an individual. Through his constant encouragement and feedback, even under time-critical and demanding circumstances, I have improved my research abilities, soft skills, and perseverance. The opportunities he created for me—from presenting at conferences to building networks and gaining recognition—have shaped my career trajectory. I will always carry the inspirations and lessons from Prof. Pasricha as I continue my journey as a researcher.

I am also deeply thankful to the respected members of my Ph.D. committee: Prof. Anthony Maciejewski, Prof. Anura Jayasumana, Prof. Nikhil Krishnaswamy, and Prof. Chuck Anderson. Their thoughtful feedback and insightful suggestions helped me rediscover new aspects of my research and refine my work from multiple perspectives.

My gratitude also extends to my mentors at Xilinx (now AMD) and Advanced Micro Devices Inc. (AMD), whose guidance has been pivotal in shaping my career. In particular, I want to thank my manager, Dr. Syed Shah and mentors Dr. Hai Lin, Dr. Ahmed Abulila, and Phil Park, for recognizing my contributions and fostering my professional growth at AMD.

A special thanks to my friends and colleagues at Prof. Pasricha's EPIC Lab, whose camaraderie and support made this journey truly memorable: Dr. Saideep Tiku, Dr. Asif Anwar Baig Mirza, Dr. Ebad Taheri, Sirui Qi, Kamil Khan, Salma Afifi, Akhil Singampalli, Pooja Anandathirtha, Christian Westbrook, Amin Shafiee, Amin Mahdian, Muddasir Attar Ismail, and Janaki Subramani. Your encouragement and friendship have been invaluable throughout this experience.

Finally, I am profoundly grateful to my wonderful family—my father Ashraf Shukoor, my mother Faiqua Ashraf, and my sister Hoorya Ayrene. Their unwavering love, humility, and guidance have been a constant source of strength, inspiring me to become a better person every day. Without their support, this journey would not have been possible.

TABLE OF CONTENTS

ABSTRACT.....	ii
ACKNOWLEDGEMENTS.....	vi
LIST OF TABLES.....	xx
LIST OF FIGURES	xxii
LIST OF CONFERENCE PUBLICATIONS.....	xxxii
LIST OF JOURNAL PUBLICATIONS.....	xxxiv
INTRODUCTION TO INDOOR LOCALIZATION.....	1
1.1 BACKGROUND ON INDOOR LOCALIZATION	3
1.1.1. DISTANCE-BASED METHODS: TRILATERATION	5
1.1.2. ANGLE-BASED METHODS: TRIANGULATION	7
1.1.3. FINGERPRINTING-BASED INDOOR LOCALIZATION.....	9
1.2. CHALLENGES OF FINGERPRINTING-BASED METHODS	11
1.2.1. BALANCING ACCURACY AND ENERGY EFFICIENCY.....	11
1.2.2. DEVICE HETEROGENEITY	12
1.2.3. SECURITY VULNERABILITIES AND ADVERSARIAL ATTACKS	12
1.2.4. LONG-TERM INDOOR LOCALIZATION.....	13
1.2.5. ADAPTATION TO DYNAMIC ENVIRONMENTS.....	13
1.2.6. EFFICIENT DATA MANAGEMENT.....	14
1.2.7. EXPLAINABILITY AND TRANSPARENCY	14
1.3. DISSERTATION OVERVIEW.....	15

ANVIL: SMARTPHONE INVARIANT INDOOR LOCALIZATION USING MULTI-HEADED ATTENTION NEURAL NETWORKS	20
2.1. INTRODUCTION	20
2.2. CONTRIBUTIONS	22
2.3. RELATED WORKS	23
2.4. ANALYSIS OF HETEROGENEITY	26
2.5. ATTENTION MECHANISM.....	28
2.6. ANVIL FRAMEWORK	31
2.6.1. FRAMEWORK OVERVIEW	31
2.6.2. RSS FINGERPRINT PRE-PROCESSING	32
2.6.3. FINGERPRINT AUGMENTATION (FAST) STACK.....	33
2.6.4. MULTI-HEADED ATTENTION MODEL	34
2.7. EXPERIMENTS	37
2.7.1. EXPERIMENTAL SETUP.....	37
2.7.2. COMPARISON WITH PRIOR WORKS	39
2.7.3. EXPERIMENTAL RESULTS.....	40
2.8. CONCLUSION	43
VITAL: HETEROGENEITY RESILIENT INDOOR LOCALIZATION USING VISION TRANSFORMER NEURAL NETWORKS	45
3.1. INTRODUCTION	45
3.2. CONTRIBUTIONS	47
3.3. RELATED WORK	48
3.4. VISION TRANSFORMERS OVERVIEW	50

3.5. VITAL FRAMEWORK	53
3.5.1. DATA AUGMENTATION MODULE (DAM)	54
3.5.2. VISION TRANSFORMER	55
3.6. EXPERIMENTS	58
3.6.1. EXPERIMENTAL SETUP	58
3.6.2. HYPERPARAMETER EXPLORATION	60
3.6.3. COMPARISON WITH STATE-OF-THE-ART	62
3.6.4. EFFECTIVENESS OF DAM	64
3.6.5. EXPERIMENTS ON NEW SMARTPHONES	65
3.7. CONCLUSION	67
STELLAR: SIAMESE MULTI-HEADED ATTENTION NEURAL NETWORKS FOR OVERCOMING TEMPORAL VARIATIONS AND DEVICE HETEROGENEITY WITH INDOOR LOCALIZATION	68
4.1. INTRODUCTION	68
4.2. CONTRIBUTIONS	70
4.3. RELATED WORK	72
4.5. ANALYSIS OF TEMPORAL VARIATIONS	77
4.6. STELLAR FRAMEWORK	79
4.6.1. RSS FINGERPRINT PRE-PROCESSING	81
4.6.2. TRIPLET SELECTION FOR CONTRASTIVE LEARNING	81
4.6.3. ANCHOR TRIPLET SELECTION	82
4.6.4. POSITIVE TRIPLET SELECTION	82
4.6.5. NEGATIVE TRIPLET SELECTION	83

4.6.6. SIAMESE MULTI-HEADED ATTENTION NEURAL NETWORK	84
4.6.7. MULTI-HEADED ATTENTION NEURAL NETWORK	85
4.6.8. FINGERPRINT AUGMENTATION STACK (FAST LAYER)	88
4.6.9. TRIPLET LOSS.....	89
4.6.10. EXTREME GRADIENT BOOSTING (XGBOOST)	90
4.7. EXPERIMENTS	91
4.7.1. DATA COLLECTION AND PRE-PROCESSING	92
4.8. EXPERIMENTAL RESULTS.....	96
4.8.1. HYPERPARAMETER TUNING: AP DROPOUT.....	96
4.8.2. ABLATION STUDY: POST ENCODING CLASSIFIERS	98
4.8.3. ABLATION STUDY: SENSITIVITY OF TRAINING DATA.....	100
4.8.4. COMPARISON VS. STATE-OF-THE-ART (HETEROGENEITY)	102
4.8.5. COMPARISON VS. STATE-OF-THE-ART (TEMPORAL VARIATIONS)	105
4.8.6. EVALUATION ON EXTENDED DEVICES AND TIME PERIOD.....	110
4.9. CONCLUSIONS.....	112
SANGRIA: STACKED AUTOENCODER NEURAL NETWORKS WITH GRADIENT	
BOOSTING FOR INDOOR LOCALIZATION	113
5.1. INTRODUCTION	113
5.2. CONTRIBUTIONS	115
5.3. RELATED WORK	115
5.4. SANGRIA FRAMEWORK	117
5.4.1. DATA COLLECTION, PREPROCESSING, AND DATABASE CREATION	118
5.4.2. GREEDY LAYER-WISE STACKED AUTOENCODER	118

5.4.3. CATEGORICAL GRADIENT BOOSTING.....	120
5.5. EXPERIMENTS	121
5.5.1. EXPERIMENTAL RESULTS.....	122
5.5.2. EFFECT OF THE STACKED AUTOENCODER.....	125
5.5.3. MODEL INFERENCE TIME LATENCY	126
5.6. CONCLUSION	127
FEDHIL: HETEROGENEITY RESILIENT FEDERATED LEARNING FOR ROBUST	
INDOOR LOCALIZATION WITH MOBILE DEVICES.....	128
6.1. INTRODUCTION	129
6.2. CONTRIBUTIONS	132
6.3. RELATED WORK.....	133
6.4. HETEROGENEITY ANALYSIS OF WI-FI RSS FINGERPRINTS	136
6.5. BACKGROUND ON FEDERATED LEARNING (FL) FOR INDOOR LOCALIZATION	
.....	138
6.5.1. FEDERATED AVERAGING (FEDAVG).....	140
6.5.2. FEDERATED STOCHASTIC GRADIENT DESCENT (FEDSGD).....	142
6.6. FEDHIL FRAMEWORK.....	143
6.6.1. GLOBAL MODEL (GM) ARCHITECTURE.....	144
6.6.2. CUSTOM STACKED AUTOENCODER (SAE) NEURAL NETWORK.....	145
6.6.3. SHALLOW NEURAL NETWORK (SNN)	147
6.6.4. LOCAL MODEL (LM) ARCHITECTURE.....	148
6.6.5. FEDHIL AGGREGATION TECHNIQUE	149
6.7. EXPERIMENTAL SETUP	152

6.7.1. SENSITIVITY ANALYSIS FOR FEDHIL	156
6.7.2. CLIENT-SERVER COMMUNICATION LATENCY	159
6.7.3. IMPACT OF CUSTOM SAE IN FEDHIL.....	160
6.7.4. COMPARISON WITH PRIOR FL AND NON-FL INDOOR LOCALIZATION FRAMEWORKS	164
6.7.5. SKEWNESS ANALYSIS.....	169
6.7.6. SCALABILITY ANALYSIS.....	171
6.8. CONCLUSION	172
CALLOC: CURRICULUM ADVERSARIAL LEARNING FOR SECURE AND ROBUST INDOOR LOCALIZATION	173
7.1. INTRODUCTION	173
7.2. CONTRIBUTIONS	176
7.3. RELATED WORK	176
7.4. ADVERSARIAL ATTACK FORMULATION.....	178
7.5. ADVERSARIAL ATTACK MECHANISM	179
7.5.1. WHITE-BOX ADVERSARIAL ATTACK FORMULATION	181
7.5.2. ADVERSARIAL DATA GENERATION	183
7.6. CALLOC FRAMEWORK.....	184
7.6.1. CURRICULUM SELECTION	185
7.6.2. LOWER-DIMENSIONAL HYPERSPACE.....	186
7.6.3. SCALED DOT-PRODUCT ATTENTION NEURAL NETWORK.....	187
7.6.4. ADAPTIVE CURRICULUM.....	188
7.7. EXPERIMENTAL SETUP	189

7.7.1. CALLOC EVALUATIONS: DEVICES, FLOORPLANS AND ATTACKS	190
7.7.2. EVALUATING IMPACT OF CURRICULUM LEARNING WITH ϵ	192
7.7.3. COMPARING CALLOC AGAINST STATE-OF-THE-ART.....	193
7.7.4. EVALUATING IMPACT OF VARYING $\emptyset$	194
7.8. CONCLUSION	195
SENTINEL: SECURING INDOOR LOCALIZATION AGAINST ADVERSARIAL ATTACKS	
WITH CAPSULE NEURAL NETWORKS	196
8.1. INTRODUCTION.....	197
8.2. CONTRIBUTIONS	200
8.3. RELATED WORK	201
8.4. ADVERSARIAL ATTACKS IN INDOOR LOCALIZATION.....	205
8.4.1. ROGUE AP ATTACK IMPLEMENTATION.....	207
8.4.2. ADVERSARIAL ATTACK METHODS.....	209
8.4.3. ADVERSARIAL ATTACK FORMULATION FOR ML INDOOR LOCALIZATION	212
8.5. SENTINEL FRAMEWORK: OVERVIEW.....	213
8.5.1. ADVERSARIAL TRAINING MECHANISM.....	215
8.5.2. FINGERPRINT IMAGE GENERATION.....	216
8.5.3. CAPSULE NEURAL NETWORK ARCHITECTURE	217
8.6. EXPERIMENTAL SETUP	221
8.6.1. EFFECTS OF ADVERSARIAL TRAINING ON HETEROGENEITY	223
8.6.2. EVALUATING THE IMPACT OF VARYING COMPROMISED APS (Φ).....	227
8.6.3. EVALUATING THE IMPACT OF VARYING PERTURBATIONS (ϵ)	228
8.6.4. COMPARISON AGAINST STATE-OF-THE-ART FRAMEWORKS	230

8.6.5. EVALUATION ON THE NEW REAL-WORLD ROGUE AP ATTACK DATASET ..	232
8.7. CONCLUSION	234
SAFELOC: OVERCOMING DATA POISONING ATTACKS IN HETEROGENEOUS	
FEDERATED MACHINE LEARNING FOR INDOOR LOCALIZATION	236
9.1. INTRODUCTION	236
9.2. CONTRIBUTIONS	239
9.3. RELATED WORK	240
9.4. DATA POISONING ATTACKS IN FL INDOOR LOCALIZATION	242
9.5. DATA POISONING ATTACK METHODS	244
9.6. SAFELOC FRAMEWORK	247
9.6.1. FUSED NEURAL NETWORK ARCHITECTURE	248
9.6.2. SALIENCY-MAP BASED AGGREGATION.....	250
9.7.1. DETERMINATION OF RECONSTRUCTION THRESHOLD (T)	253
9.7.2. EFFECTS OF DATA POISONING ATTACKS IN SAFELOC.....	254
9.7.3. COMPARISION WITH STATE-OF-THE-ART	256
9.7.4. SCALABILITY ANALYSIS WITH INCREASING NUMBER OF CLIENTS	257
9.8. CONCLUSIONS.....	258
CIELO: CLASS-INCREMENTAL CONTINUAL LEARNING FOR OVERCOMING	
CATASTROPHIC FORGETTING WITH SMARTPHONE-BASED INDOOR	
LOCALIZATION	259
10.1. INTRODUCTION	259
10.2. CONTRIBUTIONS	263
10.3. RELATED WORK	264

10.4. CONTINUAL LEARNING: OVERVIEW	266
10.4.1. CIL IN INDOOR LOCALIZATION.....	267
10.4.2. CATASTROPHIC FORGETTING	269
10.5. CIELO FRAMEWORK	270
10.5.1. REPRESENTATION MEMORY MANAGEMENT (RMM)	271
10.6. EXPERIMENTAL SETUP	276
10.6.1. SENSITIVITY ANALYSIS: DETERMINING SAMPLE SELECTION SIZE (K).....	277
10.6.2. SENSITIVITY ANALYSIS: DETERMINING MEMORY BUDGET (<i>MB</i>)	278
10.6.3. COMPARISONS WITH STATE-OF-THE-ART	280
10.6.4. IMPACT OF NEW CLASS INCREMENT SIZE ON PERFORMANCE.....	284
10.6.5. EVALUATION OF COMPUTATIONAL COST ON EDGE SERVER.....	286
10.7. CONCLUSION	287
DAILOC: DOMAIN-INCREMENTAL LEARNING FOR INDOOR LOCALIZATION USING SMARTPHONES	288
11.1. INTRODUCTION	288
11.2. CONTRIBUTIONS	291
11.3. RELATED WORK	291
11.4. BACKGROUND: DOMAIN INCREMENTAL LEARNING FOR INDOOR LOCALIZATION	293
11.5. DAILOC FRAMEWORK.....	295
11.5.1. DISENTANGLED REPRESENTATION ADAPTATION	297
11.5.1. CLASS LATENT ALIGNMENT	299
11.5.3. JOINT DOMAIN-AWARE OPTIMIZATION	300

11.6. EXPERIMENTAL SETUP	301
11.6.1. COMPARISON WITH STATE-OF-ART	303
11.6.2. DAILOC PERFORMANCE ANALYSIS	305
11.6.3. ABLATION STUDY: IMPACT OF DISENTANGLEMENT	307
11.6.4. ABLATION STUDY: IMPACT OF DEVICE ONBOARDING ORDER	308
11.7. CONCLUSION	309
GATE: GRAPH ATTENTION NEURAL NETWORKS WITH REAL-TIME EDGE	
CONSTRUCTION FOR ROBUST INDOOR LOCALIZATION USING MOBILE	
EMBEDDED DEVICES	310
12.1. INTRODUCTION	311
12.2. CONTRIBUTIONS	316
12.3. RELATED WORK	317
12.4. REPRESENTING FINGERPRINT VECTORS AS A GRAPH FOR INDOOR	
LOCALIZATION	320
12.4.1. STRUCTURAL ANALYSIS OF WI-FI RSS FINGERPRINT VECTORS: EUCLIDEAN	
VS. NON-EUCLIDEAN REPRESENTATION	320
12.4.2. REPRESENTING FINGERPRINT VECTORS AS A GRAPH	322
12.4.3. EDGE CONSTRUCTION AND MESSAGE PASSING FOR FINGERPRINT	
VECTORS	324
12.4.4. EUCLIDEAN DISTANCE-BASED MESSAGE PASSING	325
12.4.5. K-NEAREST NEIGHBOR (KNN)-BASED MESSAGE PASSING	326
12.4.6. ATTENTION-BASED MESSAGE PASSING	327
12.5. GATE FRAMEWORK FOR ROBUST INDOOR LOCALIZATION	328

12.5.1. GATE ATTENTION SCORE MECHANISM.....	330
12.5.2. MULTI-DIMENSIONAL HYPERSPACE VECTOR (MDHV) COMPUTATION	332
12.5.3. REAL-TIME EDGE CONSTRUCTION (RTEC) ALGORITHM.....	333
12.6. EXPERIMENTAL SETUP	335
12.6.1. SENSITIVITY ANALYSIS: GCN COMPRESSION SIZE AND NUMBER OF EDGES PER NODE	337
12.6.2. SENSITIVITY ANALYSIS: EFFECTS OF FINGERPRINT SAMPLES PER RP ON HETEROGENEITY RESILIENCE.....	342
12.6.3. COMPONENT-WISE ANALYSIS OF GATE: IMPACT ON WORST-CASE ERRORS AND HETEROGENEITY RESILIENCE	344
12.6.4. IMPACT OF VARYING AP DENSITIES ON GATE VARIANTS: PERFORMANCE ACROSS TRUNCATED FINGERPRINT VECTOR SIZE	346
12.6.5. COMPARISON OF GATE VARIANTS WITH STATE-OF-THE-ART INDOOR LOCALIZATION FRAMEWORKS.....	348
12.7. CONCLUSIONS.....	353
TOWARDS EXPLAINABLE INDOOR LOCALIZATION: INTERPRETING NEURAL NETWORK LEARNING ON WI-FI FINGERPRINTS USING LOGIC GATES.....	354
13.1. INTRODUCTION	354
13.2. CONTRIBUTIONS	356
13.3. RELATED WORKS	357
13.4. INTERPRETING DL FOR INDOOR LOCALIZATION.....	359
13.4.1. STRUCTURAL ANALYSIS OF WI-FI RSS FINGERPRINTS	359
13.4.2. LOGIC GATES FOR INDOOR LOCALIZATION.....	361

13.5. THE LOGNET FRAMEWORK	363
13.6. EXPERIMENTAL SETUP	367
13.6.1. ROBUSTNESS TO TEMPORAL VARIATIONS: LOGNET VS. DNN.....	368
13.6.2. LATENT SPACE INTERPRETABILITY AND FAILURE TRACEABILITY	370
13.6.3. COMPARISON AGAINST PRIOR WORKS	373
13.7. CONCLUSIONS.....	375
DISSERTATION CONCLUSIONS AND FUTURE WORK SUGGESTIONS	376
1.1. DISSERTATION CONCLUSIONS	376
1.2. SUGGESTIONS FOR FUTURE WORK	381
BIBLIOGRAPHY.....	386

LIST OF TABLES

Table 1: Details of smartphones used in experiments.	38
Table 2: Smartphones used for evaluation (Base devices).	60
Table 3: Smartphones used for evaluation (Extended devices).	66
Table 4: Details of smartphones used in our experiments.	92
Table 5: Details of extended smartphones used in our experiments.	110
Table 6: Trainable parameters for each stack in the SAE in SANGRIA.	122
Table 7: Avg. inference time (milliseconds) across devices and frameworks.	126
Table 8: Trainable parameters in each layer in the custom SAE.	152
Table 9: Trainable parameters in each layer in the SNN model.	153
Table 10: Details of smartphones used in our experiments.	154
Table 11: Inference time latency for the SNN model on different smartphones in milli-seconds (ms).	155
Table 12: Smartphones details used to evaluate the CALLOC framework.	189
Table 13: Building floorplan details used to evaluate the CALLOC framework.	190
Table 14: Devices used to collect RSS fingerprints.	221
Table 15: Model parameters, size of all frameworks.	231
Table 16 : Rogue AP devices used in rssrogueloc.	233
Table 17 : model latency and parameters comparison.	257
Table 18: CPU usage, average power, and mean localization error comparison across frameworks	287
Table 19: Details of smartphones used in experiments	303

Table 20: Comparison of hardware performance metrics (averaged across all seven devices and five buildings) for GATE and baselines.	351
Table 21: Truth Table of logic gates.....	361
Table 22: Comparison of model parameters and model latency.....	374

LIST OF FIGURES

Figure 1: Real-world applications of indoor localization.	1
Figure 2: Global indoor localization market size in 2025 and projected market size growth.	2
Figure 3: A graphical representation of the trilateration indoor localization process.....	5
Figure 4: A graphical representation of triangulation-based indoor localization.	8
Figure 5: A graphical representation of fingerprinting-based indoor localization.	10
Figure 6: <i>SPARK</i> overview: challenges addressed (blue), corresponding <i>SPARK</i> frameworks (green), and underlying ML algorithms (yellow).	16
Figure 7: RSS values for Wi-Fi APs as simultaneously observed by two unique smartphones at a specific location. Solid lines represent the mean values. Shaded regions represent the range of RSS values, over 10 fingerprint readings. Distinct Wi-Fi APs (by MAC IDs) are denoted as unique integers from 0 to 50.	27
Figure 8: A procedural representation of scaled-dot-product attention (left) and multi-head attention (right). Each attention layer accepts three inputs: Query (Q), Keys (K), and Values (V).	30
Figure 9: An overview of the ANVIL indoor localization framework depicting the offline (red arrows) and online (green arrows) phases.	31
Figure 10: Various components of the Fingerprint Augmentation Stack (FASt). Each subcomponent augments a specific heterogeneity effect.....	33
Figure 11: Overview of the multi-head attention neural network model in the <i>ANVIL</i> framework.	35

Figure 12: Indoor floorplans within the benchmark suite. Reference points are annotated with yellow boxes.	37
Figure 13: The mean localization error in meters for the framework ANVIL proposed in [22] as compared to state-of-art previous works.....	38
Figure 14: The localization performance of <i>ANVIL</i> as compared to state-of-the-art previous works.	43
Figure 15: Overview of Vision Transformer Neural Network	51
Figure 16: An overview of the proposed device heterogeneity-resilient VITAL indoor localization framework.	54
Figure 17: Benchmark indoor paths that were used for fingerprint collection. We collected RSS data across four buildings, with varying path lengths. The blue dots indicate reference points (RP) at which the RSS reading were taken. Six different smartphones were used to capture RSS data at each RP.	59
Figure 18: Surface plot showing the impact of patch size and RSS image sizes on the mean indoor localization error.	60
Figure 19: Impact of number of fine tuning MLP layers and number of MSA heads in our vision transformer model on mean indoor localization error.	61
Figure 20: Min (lower whisker), Mean (red bar), and Max (upper whisker) error across all buildings for comparison frameworks w/ base devices	64
Figure 21: Slope graph of the impact of DAM on all frameworks.	65
Figure 22: Min (lower whisker), Mean (red bar), and Max (upper whisker) error across all buildings for comparison frameworks w/ extended devices.....	66
Figure 23: Analysis of device heterogeneity, across different smartphone at the same time.....	76

Figure 24: Analysis of temporal variations across varying time periods as captured by the same smartphone at the same location.	77
Figure 25: Working of the STELLAR framework, including offline (training) phase	79
Figure 26: The algorithm of the Multi-headed attention mechanism.	86
Figure 27: Floorplan layout for Building 1 (64 meters long with 64 RPs) and Building 2 (62 meters long with 62 RPs) and RP (blue dots).....	93
Figure 28: Analyzing visible (green) APs and invisible (red) APs over time.	94
Figure 29: Tuning D% of AP dropout during offline phase for both building floorplans with min-max intervals showing best and worst-case errors for each D% value.....	98
Figure 30: Comparing response of different classifier in temporal space for both building floorplans. Blue shadow indicates best and worst-case errors for STELLAR.	99
Figure 31: Sensitivity analysis of training data on STELLAR's performance. Blue shadow indicates best and worst-case errors for STELLAR.	102
Figure 32: Framework comparison: average localization errors (meters) across devices.	103
Figure 33: Average errors across training - testing devices from both building floorplans with min-max intervals showing best and worst-case errors for each framework.	103
Figure 34: STELLAR's performance analysis: average localization error (meters) across training and testing devices with varied temporal variations (CIs).....	105
Figure 35: Difference between best and worst performing testing device per training device per CI.....	107
Figure 36: Average errors across all frameworks in the temporal space, blue shadow indicates best and worst-case errors for STELLAR.	108

Figure 37: Box plot illustrating the minimum, maximum, and mean error differences between STELLAR and other frameworks per CI.....	109
Figure 38: Average errors across all frameworks in the temporal space for extended devices with min-max intervals showing best and worst-case errors for each framework.....	111
Figure 39: Implementation overview of the SANGRIA framework	117
Figure 40: Greedy layer-wise stacked autoencoder (SAE) architecture with Encoder (gray), Decoder (green), and bottleneck (orange) layers.	119
Figure 41: Min (lower whisker), Mean (green bar), and Max (upper whisker) error across all comparison frameworks for indoor localization.	123
Figure 42: Experimental results for the top four best performing frameworks with average errors (meters) across all floorplans.	124
Figure 43: Mean localization errors across all frameworks using UJI dataset	125
Figure 44: Bar plot comparing average errors of all frameworks with SAE (Blue) and without SAE (Orange)	126
Figure 45: RSS fingerprints of visible Wi-Fi APs collected using four different smartphones at the same indoor location.	137
Figure 46: Federated learning (FL) overview using multiple client devices for indoor localization.	140
Figure 47: GM architecture for FedHIL	144
Figure 48: LM architecture for FedHIL.....	148
Figure 49: Sensitivity Analysis of RP Locations in Building 5 Floorplan for Selecting H Value in FedHIL Technique	157
Figure 50: Average error in meters for different H value in FedHIL technique.....	158

Figure 51: Average communication latency in seconds for different H value in FedHIL technique.	159
Figure 52: FedHIL with different SAE configurations, training device: MOTO (values in cells represent Euclidean distance).	161
Figure 53: Impact of Custom SAE on Average Error in Building Floorplans	163
Figure 54: Comparison of Original Fingerprint and Augmented Fingerprint Using Custom SAE on Device: OP3	164
Figure 55: Average localization across all smartphones and floorplans for comparison with FL and non-FL frameworks	165
Figure 56: Device- specific performance across all building floorplans and FL frameworks for different training devices.	167
Figure 57: Susceptibility to noise across FL frameworks with training device : MOTO, testing device OP3 in Building 5.....	168
Figure 58: Skewness analysis on different floorplans and devices with different device mixes among clients.	170
Figure 59: Scalability analysis on different floorplans and varying devices combinations split.	171
Figure 60: Accuracy reduction (increase in localization error) in three ML-based indoor localization solutions [17], [100] due to adversarial attacks.....	175
Figure 61: Illustration of indoor localization system with and without adversarial attacks – A:1 (weak attack), A:2 (strong attack).....	179
Figure 62: Overview of the CALLOC framework: offline and online phases	184
Figure 63: Localization errors across different devices, buildings, and attacks	191
Figure 64: Impact of curriculum learning on different attacks and ϵ	192

Figure 65: Comparing CALLOC against state-of-the-art frameworks.....	193
Figure 66: Effect of varying attacked APs (ϕ) on localization error.....	194
Figure 67: Impact of rogue APs on three popular ML-based indoor localization solutions [119], [120], [121] from prior work.	199
Figure 68: Spatial invariance problem in deep learning algorithms. Both cases are classified as valid human faces by a CNN model.	203
Figure 69: RSS fluctuations in indoor environment depicting real-world scenarios with and without the presence of rogue APs.....	205
Figure 70: Evil twin attack during indoor localization.	208
Figure 71: Man-in-the-middle attack during indoor localization.	209
Figure 72: Overview of the SENTINEL framework, including the offline (training) phase and online (inference) phase.	215
Figure 73: Building floorplan layouts with varying path length, visible APs, and characteristics.	223
Figure 74: Performance of the SENTINEL variants across different devices and building floorplans.	225
Figure 75: Performance summary for SENTINEL variants.	226
Figure 76: Performance of the four SENTINEL variants on simulated adversarial attacks through varying ϕ	227
Figure 77: Performance of the three SENTINEL variants on simulated adversarial attacks through varying ϵ	230
Figure 78: Performance comparisons of all SENTINEL variants against state-of-the-art indoor localization frameworks.....	230

Figure 79: Device-Floorplan wise performance of SENTINEL-FGSM on RSSRogueLoc dataset.	232
Figure 80: Performance comparisons of all SENTINEL models against state-of-the-art on the RSSRogueLoc dataset.	234
Figure 81: Indoor localization error comparison for FEDLOC [175] and FEDHIL [88] under label flipping and backdoor poisoning attacks.	238
Figure 82: Working of an FL-based indoor localization framework with malicious clients (with backdoor and label-flipping attacks).	243
Figure 83: Fused neural network architecture used in SAFELOC.	250
Figure 84: Impact of varying the reconstruction threshold (τ) on mean localization error across five building floorplans.	254
Figure 85: Mean localization error under various data poisoning attacks and perturbation magnitudes (ϵ).	254
Figure 86: Comparison of SAFELOC against state-of-the-art.	255
Figure 87: Mean localization error with increasing number of (total, poisoned) clients ratio.	257
Figure 88: Illustration of catastrophic forgetting using a four-layer DNN. (a) DNN predictions after training on initial 30 RPs (0 to 30). (b) DNN predictions after retraining exclusively on 10 new RPs (31 to 40).	262
Figure 89: Overview of the CIELO framework for class incremental learning in Wi-Fi RSS fingerprinting-based indoor localization.	270
Figure 90: Confidence and sensitivity measure for memory management.	271
Figure 91: Mean localization error with increasing number of samples per class.	278
Figure 92: Mean localization error with increasing Mb	279

Figure 93: Comparison of mean localization errors across different state-of-the-art frameworks under both self and cross-device testing for Building 1.	281
Figure 94: Comparison of mean localization error across different state-of-the-art frameworks under both self and cross-device testing for Building 2.	282
Figure 95: Comparison of CIELO against the state-of-the-art.	284
Figure 96: Mean localization error with varying number of new classes (RPs) introduced through retraining.	285
Figure 97: Average retraining loss with varying number of new classes (RPs) introduced through retraining.	286
Figure 98: Illustration on impact of domain shifts arising from device heterogeneity and temporal variations on indoor localization performance.	290
Figure 99: Overview of the DAILOC framework using domain-aware incremental learning for robust Wi-Fi RSS fingerprinting-based indoor localization.	295
Figure 100: DAILOC’s multi-level VAE with disentanglement representation.	297
Figure 101: Statistical alignment of class embeddings using MMD loss to prevent catastrophic forgetting, as part of the CESA module.	300
Figure 102: Comparison of DAILOC against state-of-the-art.	305
Figure 103: Comparison of mean localization error (in meters) for Building 1.	306
Figure 104: Comparison of mean localization error (in meters) for Building 2.	306
Figure 105: Mean pseudo-label error before and after disentanglement across time in two buildings.	308
Figure 106: Mean localization error (in meters) across different heterogeneous device introduction orderings over time.	309

Figure 107: Localization performance comparison of GNN [233] against state-of-the-art ML methods for a 60-meter-long indoor environment. ML models evaluated include an Attention-based Neural Network [45], a fully connected Deep Neural Network (DNN) [42], Gaussian Process Classification (GPC) [17], a Convolutional Neural Network (CNN) [6], and K-Nearest Neighbor (KNN) [229].	314
Figure 108: Fingerprint vector with Euclidean noise assumption (left) vs. real-world non-Euclidean structure (right).	322
Figure 109F: Graph representation of fingerprint vectors (left) and message passing between nodes (right).	324
Figure 110: Overview of the GATE framework during offline training and online inference phases.	330
Figure 111: Overview of the RTEC algorithm during the online phase.	333
Figure 112: Spatial layouts of the building floorplans (left) and mobile device specifications (right).	335
Figure 113: Sensitivity of GATE to GCN compression size (H%) and edge connectivity (NB%) for each building floorplan.	338
Figure 114: Impact of edge connectivity (NB%) on GATE's localization accuracy, inference latency, and energy-delay product (EDP) across all mobile devices and buildings.	340
Figure 115: Sensitivity of GATE to the number of fingerprint samples per RP during training.	344
Figure 116: Component-wise analysis of different GATE variants.	346
Figure 117: Impact of varying AP densities (fingerprint vector truncation) on localization accuracy across GATE variants.	347

Figure 118: Evaluation of the GATE variants compared to state-of-the-art indoor localization frameworks.	350
Figure 119: Euclidean vs. Non-Euclidean fingerprint structures.....	360
Figure 120: Temporal variation of RSS values across the first 10 APs.....	360
Figure 121: The LogNet and DNN-DownSample architectures.....	364
Figure 122: Mean localization error across six LogNet variants and DNN-DownSample at CI:0 (without temporal variations).....	369
Figure 123: Mean localization error across six LogNet variants and DNN-DownSample over temporal variations (CI:0-9).	370
Figure 124: Latent space encodings of LogNet-NOR and DNN-DownSample on Building 1 across all RPs at CI:0.	373
Figure 125: Impact of varying hidden layers across LogNet and DNNs.....	374

LIST OF CONFERENCE PUBLICATIONS

- Saideep Tiku, Danish Gufran, and Sudeep Pasricha. "Multi-head attention neural network for smartphone invariant indoor localization." IEEE international conference on indoor positioning and indoor navigation (IPIN), 2022. (Best Paper Award)
- Danish Gufran, Saideep Tiku, and Sudeep Pasricha. "VITAL: Vision transformer neural networks for accurate smartphone heterogeneity resilient indoor localization." ACM/IEEE Design Automation Conference (DAC), 2023.
- Danish Gufran and Sudeep Pasricha. "FedHIL: Heterogeneity resilient federated learning for robust indoor localization with mobile devices." International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2023.
- Danish Gufran and Sudeep Pasricha. "CALLOC: Curriculum adversarial learning for secure and robust indoor localization." ACM/IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE), 2024.
- Danish Gufran, Pooja Anandathirtha, and Sudeep Pasricha. "SENTINEL: Securing Indoor Localization against Adversarial Attacks with Capsule Neural Networks." International Conference on Compilers, Architectures, and Synthesis for Embedded Systems (CASES), 2024.
- Akhil Singampalli, Danish Gufran, and Sudeep Pasricha. "SAFELOC: Overcoming Data Poisoning Attacks in Heterogeneous Federated Machine Learning for Indoor Localization." IEEE/ACM Design, Automation & Test in Europe Conference (DATE), 2024.

- Akhil Singampalli, Danish Gufran, and Sudeep Pasricha. "DAILOC: Domain-Incremental Learning for Indoor Localization using Smartphones." IEEE international conference on indoor positioning and indoor navigation (IPIN), 2025.
- Danish Gufran and Sudeep Pasricha. "GATE: Graph Attention Neural Networks with Real-Time Edge Construction for Robust Indoor Localization using Mobile Embedded Devices. " International Conference on Compilers, Architectures, and Synthesis for Embedded Systems (CASES), 2025. (Best Paper Candidate)
- Danish Gufran and Sudeep Pasricha. "Towards Explainable Indoor Localization: Interpreting Neural Network Learning on Wi-Fi Fingerprints Using Logic Gates." IEEE international conference on indoor positioning and indoor navigation (IPIN), 2025. (Best Paper Candidate)

LIST OF JOURNAL PUBLICATIONS

- Danish Gufran, Saideep Tiku, and Sudeep Pasricha. "SANGRIA: Stacked autoencoder neural networks with gradient boosting for indoor localization." IEEE Embedded Systems Letters (ESL), 2023.
- Danish Gufran, Saideep Tiku, and Sudeep Pasricha. "STELLAR: Siamese multiheaded attention neural networks for overcoming temporal variations and device heterogeneity with indoor localization." IEEE Journal of Indoor and Seamless Positioning and Navigation (ISPIN-J), 2023.
- Danish Gufran and Sudeep Pasricha. "FedHIL: Heterogeneity resilient federated learning for robust indoor localization with mobile devices." ACM Transactions on Embedded Computing Systems (TECS), 2023.
- Danish Gufran, Pooja Anandathirtha, and Sudeep Pasricha. "SENTINEL: Securing Indoor Localization against Adversarial Attacks with Capsule Neural Networks." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2024.
- Akhil Singampalli, Danish Gufran, and Sudeep Pasricha. "CIELO: Class-Incremental Continual Learning for Overcoming Catastrophic Forgetting with Smartphone-based Indoor Localization." IEEE Access, 2025.
- Danish Gufran and Sudeep Pasricha. "GATE: Graph Attention Neural Networks with Real-Time Edge Construction for Robust Indoor Localization using Mobile Embedded Devices. ACM Transactions on Embedded Computing Systems (TECS), 2025.

CHAPTER 1

INTRODUCTION TO INDOOR LOCALIZATION

Accurate indoor localization has become a critical component in today's hyper-connected world, enabling a wide range of applications such as indoor navigation, personalized services, asset tracking, geofencing, and emergency management [1], as illustrated in Figure 1. These use cases span diverse domains including retail, healthcare, public safety, and industrial operations, demonstrating the broad societal and commercial relevance of indoor localization systems [1], [4].

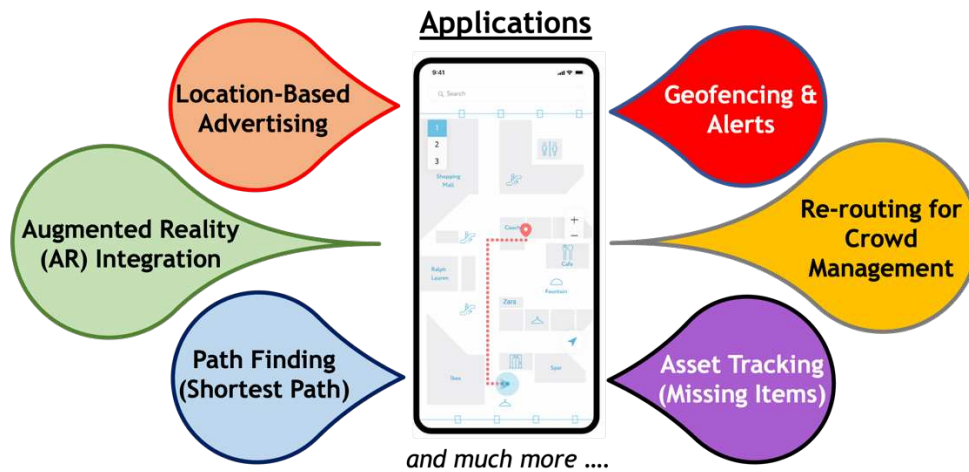


Figure 1: Real-world applications of indoor localization.

While the Global Positioning System (GPS) has transformed outdoor navigation, it remains inadequate for indoor environments due to signal attenuation caused by walls and other physical barriers [3]. This gap has created a growing demand for reliable indoor localization solutions that can function across diverse environments, including airports, hospitals, shopping malls, and factories [61]. Among the many techniques proposed, Wi-Fi-based methods stands out as a promising approach due to its ability to leverage existing Wi-Fi infrastructure for localization

without the need for additional hardware [63]. However, implementing a reliable Wi-Fi based indoor localization framework poses numerous challenges, especially when deployed on mobile devices operating in real-time and real-world dynamic environments [64]. Key concerns include achieving high localization accuracy, ensuring energy efficiency use case on mobile devices, maintaining localization system security, protecting user data privacy, and enabling adaptability across diverse mobile devices and ever-changing building layouts.

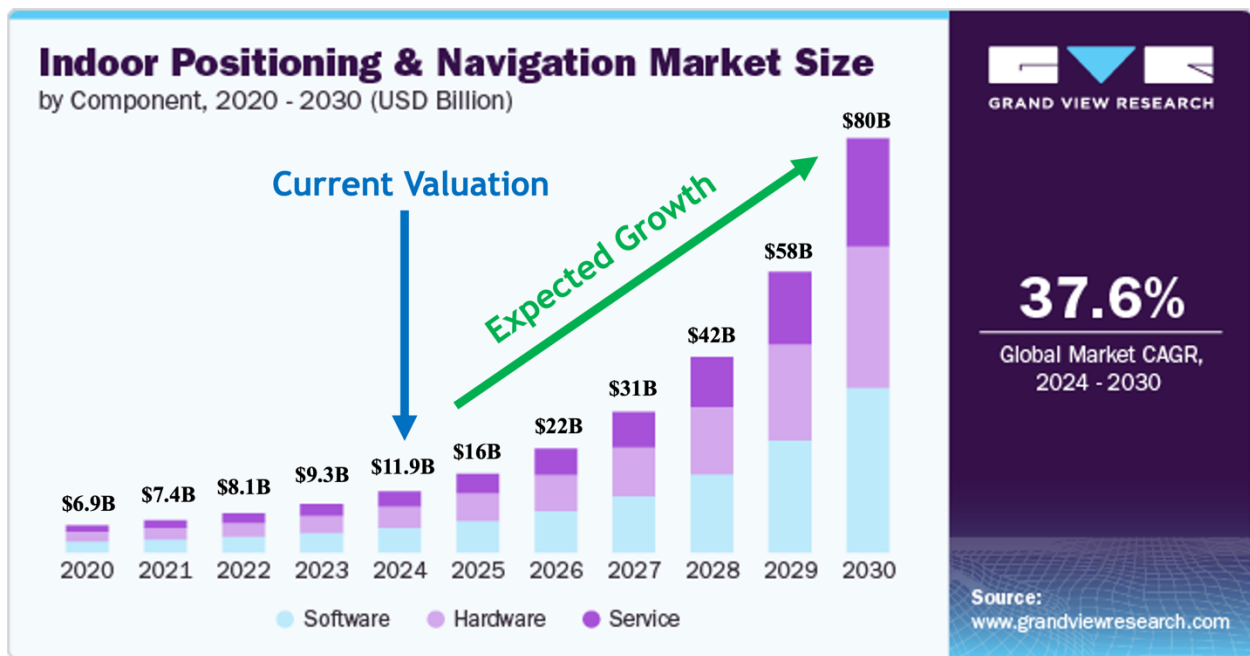


Figure 2: Global indoor localization market size in 2025 and projected market size growth.

As indoor localization transitions from research to real-world deployment, it is being actively adopted across several industries. Companies such as Apple, Google, Microsoft, HPE, Quuppa, and Zebra are investing significantly in machine learning (ML)-driven indoor localization solutions [110]. Reflecting this momentum, the global indoor localization market was valued at \$11.9 billion in 2024, with strong forecasts for continued growth. As shown in Figure 2, the market

is expected to expand substantially in the coming years, driven by the increasing demand for high-precision, intelligent, and privacy-aware localization frameworks [246].

1.1 BACKGROUND ON INDOOR LOCALIZATION

The field of indoor localization has evolved over the past few decades, driven by the need for accurate, real-time positioning in environments where GPS is ineffective. Early attempts sought to replicate outdoor navigation techniques, but the complexity of indoor spaces—characterized by obstacles, signal interference, and non-line-of-sight conditions—demanded new strategies [3]. Over time, various methods emerged, each with unique strengths and limitations, laying the foundation for modern research efforts [61], [63], [66], [70].

The first generation of indoor localization systems focused on infrared (IR) and ultrasound technologies [224]. These systems relied on specialized hardware for signal transmission and detection. While IR systems were relatively inexpensive, their strict line-of-sight requirements and sensitivity to environmental conditions limited their use. Similarly, ultrasound systems, though capable of achieving reasonable accuracy, suffered from signal interference in noisy environments, restricting their scalability. The next significant advancement came with radio frequency (RF)-based systems, which utilized wireless signals such as Wi-Fi, Bluetooth, and ultra-wideband (UWB). These systems leveraged existing wireless infrastructure, making them more practical to deploy without additional hardware [110]. However, RF-based approaches were hindered by multipath interference, where signals reflect off walls and objects, creating complex propagation patterns that negatively impacted localization accuracy.

The rise of computer vision technologies introduced camera-based localization systems [209]. These solutions used visual markers such as QR codes or applied visual feature matching to

estimate position. While offering fine-grained accuracy, vision-based systems required significant computational resources and performed poorly in low-light conditions. Privacy concerns also limited their adoption in public spaces, reducing their scalability. Another innovative approach involved magnetic field-based localization, which mapped the natural magnetic fields within buildings to serve as location fingerprints [18], [19]. Although magnetic fields are stable over time, the approach required extensive calibration and showed varying accuracy across different building structures. Similarly, inertial navigation systems (INS), using accelerometers and gyroscopes, provided indoor tracking with short-term accuracy. However, sensor drift over time restricted the use of INS for prolonged applications.

As indoor localization research matured, hybrid solutions combining multiple methods gained popularity. Sensor fusion techniques, which integrate inertial data with Wi-Fi or Bluetooth signals, emerged to address the limitations of individual approaches [149]. At the same time, machine learning (ML)-based techniques offered new possibilities by enabling systems to learn from historical data, improving adaptability to environmental changes and device variations [172]. Among the diverse indoor localization methods, trilateration [113], [252], triangulation [114], and fingerprinting [111] have proven to be highly effective. Trilateration, which estimates location by measuring distances from multiple reference points (RPs), offers reliable accuracy when precise signal distances are available. Triangulation, which uses angular information to determine position, is well-suited for environments with unobstructed access points and can complement other methods. Meanwhile, fingerprinting stands out for leveraging pre-collected signal patterns, making it resilient to multipath interference and capable of adapting to dynamic environments. Each of these methods addresses specific challenges in indoor localization, and the following sections will explore their principles, applications, and limitations in greater detail.

1.1.1. DISTANCE-BASED METHODS: TRILATERATION

Trilateration is a fundamental technique in indoor localization that estimates an object's position by measuring its distance from multiple known RPs, such as Wi-Fi access points (AP) or beacons [113], [252]. The concept behind trilateration relies on the geometric principle that the location of a point in two-dimensional (2D) or three-dimensional (3D) space can be uniquely determined when the distances from that point to at least three known RPs are measured accurately. As illustrated in Figure 3, trilateration works by representing each distance as the radius of a circle (or sphere in 3D space) centered at the known RP. The target's location is found at the intersection of these circles or spheres, provided there is no error or interference affecting the measurements.

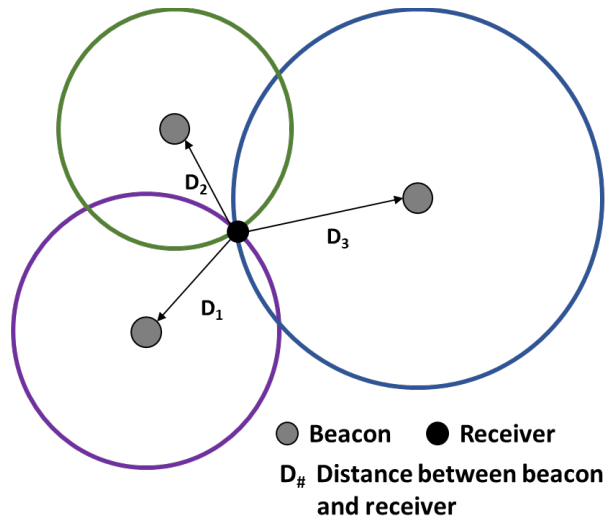


Figure 3: A graphical representation of the trilateration indoor localization process.

In a 2D space, trilateration requires the distance from three different RPs, typically measured using the Received Signal Strength (RSS) or Channel State Information (CSI) of radio waves emitted by APs or Bluetooth beacons. The RSS/CSI value decreases as the signal propagates farther, allowing the system to estimate the distance between the AP and the device. The position of the target is calculated at the point where the three circles each representing the distance from a

RP overlap. Trilateration offers several advantages, making it a popular choice for indoor localization. One key benefit is its straightforward mathematical framework, which is computationally efficient and relatively easy to implement. Unlike more complex algorithms, trilateration does not require prior knowledge of environmental conditions, which makes it adaptable across a variety of spaces. Furthermore, the method can provide reasonable accuracy in environments where the signal strengths remain consistent, and interference levels are minimal. This simplicity also allows it to integrate seamlessly with existing wireless infrastructure, using widely available APs and Bluetooth beacons for localization without requiring additional hardware installations.

However, trilateration is not without limitations. One major challenge is the accuracy of the distance estimates, which heavily depend on the reliability of the signal measurements. In real-world indoor environments, radio signals are often affected by multipath interference, where signals reflect off walls, furniture, and other objects, creating multiple paths that distort the true distance. This interference leads to errors in distance estimation, which can cause inaccuracies in location predictions. Another drawback of trilateration is its reliance on at least three RPs with clear signal transmission paths. If one or more APs are blocked or the signal is degraded due to obstacles, the localization accuracy decreases significantly. Additionally, signal attenuation caused by varying indoor conditions, such as changes in humidity or crowd density, further complicates the distance measurements, requiring periodic calibration to maintain performance. Despite these challenges, trilateration remains a valuable technique, especially in controlled environments where the infrastructure and signal conditions are stable. In applications such as asset tracking in warehouses or navigation in open indoor spaces like airports, trilateration provides a cost-effective and scalable solution. However, its effectiveness diminishes in more complex environments,

where advanced techniques, such as sensor fusion or fingerprinting, may be required to achieve higher accuracy and reliability. As discussed in the following sections, more advanced methods like triangulation and fingerprinting can address some of the limitations of trilateration, offering complementary solutions to the challenges of indoor localization.

1.1.2. ANGLE-BASED METHODS: TRIANGULATION

Triangulation is another widely used technique in indoor localization that estimates a target's position by measuring the angles at which signals (RSS/CSI) are received from known RPs [114]. Unlike trilateration, which relies on distance measurements, triangulation determines location by leveraging angle-of-arrival (AoA) or directional measurements. As illustrated in Figure 4, the method involves calculating the angles formed between the target and at least two or more RPs, such as APs, Bluetooth beacons, or UWB anchors. The intersection of the directional lines—each originating from a RP along the measured angle—pinpoints the estimated location of the target.

In a two-dimensional (2D) plane, triangulation typically uses two or more RPs with known coordinates to determine the angles at which signals arrive from the target device. These angles define directional lines from each RP, and the intersection of these lines gives the location of the target. In a three-dimensional (3D) space, more RPs or additional angular information are required to estimate the target's position accurately along all three axes. The precision of this technique hinges on the accuracy of the angle measurements and the quality of the signal paths between the reference points and the target. One of the main advantages of triangulation is its ability to provide precise localization, particularly in open environments with minimal interference. Systems utilizing high-frequency signals such as ultra-wideband (UWB) or 5G networks can achieve highly accurate angular measurements, making triangulation a practical solution for real-time localization

in well-controlled environments. Furthermore, triangulation-based systems require minimal calibration compared to data-intensive approaches such as fingerprinting, allowing them to be deployed quickly. In addition, directional antennas or phased array systems can enhance the accuracy of angle measurements, making the technique particularly effective in certain applications.

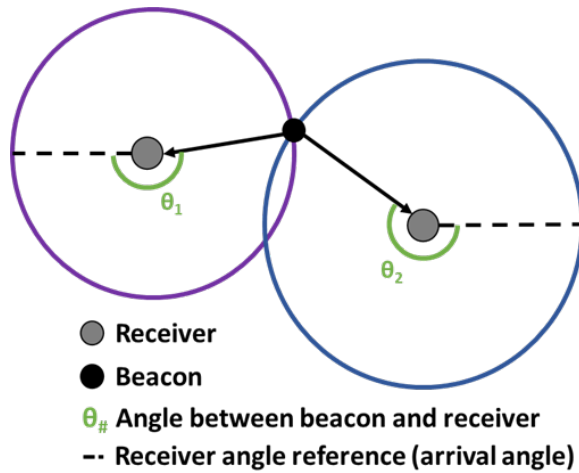


Figure 4: A graphical representation of triangulation-based indoor localization.

Despite its strengths, triangulation faces several limitations in real-world indoor environments. One significant challenge is the susceptibility of angular measurements to interference from walls, furniture, and other obstacles. These objects can reflect or block signals, leading to inaccuracies in the measured angles and consequently, the estimated position. Additionally, the precise measurement of angles often requires specialized equipment, such as phased array antennas or directional sensors, which may not be available on all mobile devices. This dependency on specialized hardware limits the applicability of triangulation in consumer-grade localization systems. Another constraint is the need for a clear line-of-sight between the target and multiple

RPs. If the signal paths are obstructed or if fewer than the required number of APs are available, the accuracy of the localization can degrade rapidly.

1.1.3. FINGERPRINTING-BASED INDOOR LOCALIZATION

Fingerprinting-based indoor localization is one of the most widely adopted techniques due to its ability to deliver high accuracy, even in complex environments with obstacles and interference [111]. Unlike distance-based and angle-based methods, fingerprinting relies on pre-constructed signal patterns rather than geometric measurements. This method works by collecting the unique characteristics of received signals, such as the Received Signal Strength (RSS), from multiple access points at various known locations within a building. These signal patterns, or “fingerprints,” are stored in a database during the offline phase. During the online phase, the current RSS values received by the user’s device are compared against the stored fingerprints to estimate the user’s location. This matching process enables precise localization, even when signals are affected by non-line-of-sight conditions and multipath interference.

As illustrated in Figure 5, the fingerprinting-based approach consists of two key phases. In the offline phase, the system captures RSS fingerprints at predefined locations throughout the indoor space. Each fingerprint is associated with a specific position and contains RSS values from visible APs. This phase is often labor-intensive, as it requires thorough coverage of the target area to build an accurate signal map. In the online phase, the user’s device collects RSS values, and the system compares these values to the stored fingerprints in real time. The location is estimated based on the closest match, either through deterministic methods, such as nearest-neighbor matching, or machine learning (ML) models [172].

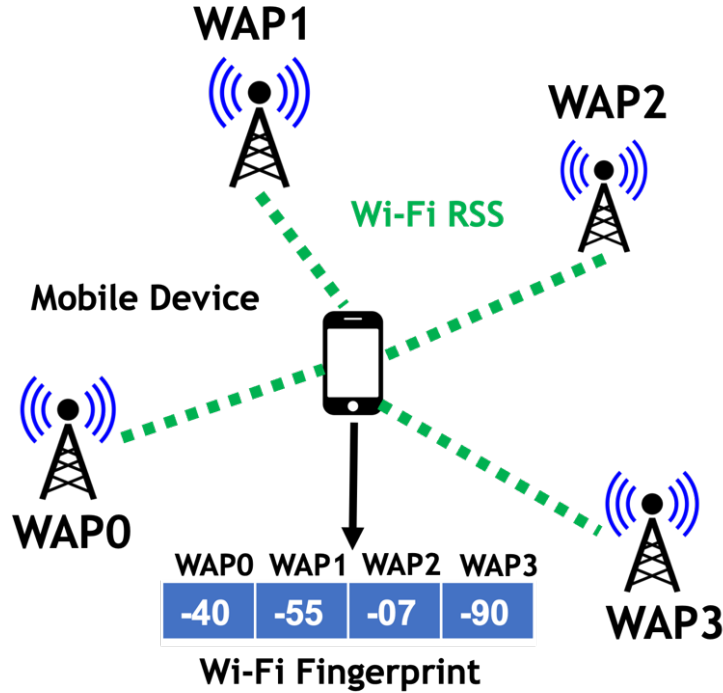


Figure 5: A graphical representation of fingerprinting-based indoor localization.

One of the significant advantages of fingerprinting is its resilience to environmental interference. Since it relies on matching signal patterns rather than precise distances or angles, fingerprinting performs well even in environments with multipath effects, where signals reflect off walls, furniture, or other objects. This makes it particularly effective in cluttered indoor spaces such as airports, shopping malls, and hospitals, where other methods struggle. Additionally, fingerprinting-based localization systems can achieve high accuracy by incorporating ML models that adapt to new signal patterns over time, improving performance in dynamic environments. However, fingerprinting-based localization has several limitations. One of the primary challenges is the effort required for data collection during the offline phase. Constructing an accurate fingerprint database demands comprehensive coverage of the area, which can be time-consuming and labor-intensive. Additionally, the system may require periodic updates to maintain accuracy,

as changes in the environment—such as rearranged furniture, varying crowd densities, or fluctuations in access point power levels—can alter the signal landscape. Without regular recalibration, the quality of the location estimates can degrade over time. We discuss the limitations of the fingerprinting method in Section 1.2 in detail.

1.2. CHALLENGES OF FINGERPRINTING-BASED METHODS

Fingerprinting-based indoor localization offers high accuracy and resilience against environmental interference, but it also presents several challenges that hinder its scalability and real-world deployment [143]. These challenges must be carefully addressed to ensure the system performs efficiently across diverse environments and devices. The following sections highlight the key issues involved in developing effective fingerprinting-based systems.

1.2.1. BALANCING ACCURACY AND ENERGY EFFICIENCY

Achieving high accuracy while maintaining energy efficiency and minimizing latency is a significant challenge in fingerprinting-based systems. Mobile devices, which are often used for localization, operate under strict resource constraints, including limited CPU power, memory, and battery life. Complex ML models can improve localization accuracy, but they also increase computational overhead, draining the battery and slowing down response times. Striking a balance between accuracy and efficiency requires the use of lightweight ML models optimized for mobile hardware, ensuring that location estimates are both reliable and fast without compromising device performance.

The trade-off between accuracy and latency is particularly crucial in real-time applications such as indoor navigation, where users expect instantaneous responses. A fingerprinting-based

system must provide precise location estimates while keeping the prediction time within acceptable limits. Techniques such as model pruning, quantization, and edge computing can help reduce computation times, allowing the system to meet real-time requirements without sacrificing too much accuracy. Ensuring that the system achieves this balance is critical to maintaining usability in practical applications.

1.2.2. DEVICE HETEROGENEITY

One of the most persistent challenges in fingerprinting-based localization is device heterogeneity. Mobile devices differ significantly in terms of hardware, such as Wi-Fi chipsets, antennas, and signal processing algorithms. These variations result in inconsistent RSS measurements, even when taken from the same location under identical conditions. As a result, the fingerprints collected with one device may not match the RSS values captured by another, reducing the accuracy of location estimates. Addressing device heterogeneity requires robust data augmentation techniques or transfer learning methods to generalize the ML model across multiple devices. Some systems also incorporate calibration procedures that align RSS values from different devices, but these procedures can be time-consuming and may not always be practical in dynamic environments. Developing ML models that can learn to adapt to different hardware configurations is essential to ensure consistent performance across various mobile platforms.

1.2.3. SECURITY VULNERABILITIES AND ADVERSARIAL ATTACKS

Fingerprinting-based localization systems are susceptible to adversarial attacks that can manipulate signal patterns and compromise the accuracy of location estimates. Jamming attacks, which interfere with Wi-Fi signals, and spoofing attacks, which falsify RSS values, can create

significant errors, leading to incorrect positioning. These attacks pose a serious threat to user privacy and localization performance, especially in sensitive applications like emergency response and asset tracking. To mitigate these risks, fingerprinting systems must incorporate specialized techniques that enable the ML model to detect and counteract malicious signals. Additionally, systems can employ signal authentication mechanisms to ensure that the RSS values originate from legitimate sources. Integrating security features into the localization framework is essential to building robust systems that can withstand intentional interference.

1.2.4. LONG-TERM INDOOR LOCALIZATION

Indoor environments are inherently dynamic, with changes in furniture layout, AP positions, and crowd densities affecting signal propagation. These variations introduce noise into the signal patterns, reducing the accuracy of fingerprint-based localization over time. Systems that rely on static fingerprints often degrade in performance unless the fingerprint database is frequently updated, which is labor-intensive and impractical for large deployments. Calibration-free ML systems that can adjust to environmental changes are essential for maintaining long-term accuracy. Techniques such as context-aware algorithms allow the system to learn from new signal patterns without manual intervention.

1.2.5. ADAPTATION TO DYNAMIC ENVIRONMENTS

Static ML models that are trained once and deployed without updates struggle to maintain accuracy in dynamic indoor environments. As new devices enter the network or APs are added or reconfigured, the signal landscape changes, rendering the initial fingerprints obsolete. Continuous learning frameworks address this issue by allowing ML models to learn incrementally from new

data, ensuring they remain accurate over time. One promising approach is FL, where the model is trained across multiple devices in a decentralized manner without sharing raw data. This approach not only preserves user privacy but also improves the adaptability of the ML model by enabling it to learn from diverse environments. Systems that can adapt in real time through continuous learning are better equipped to handle the variability of indoor spaces and deliver consistent localization performance.

1.2.6. EFFICIENT DATA MANAGEMENT

Managing the large volume of data required for fingerprinting-based localization poses significant challenges. Constructing and maintaining a comprehensive fingerprint database demands extensive effort, as it involves collecting RSS values across multiple locations and under various environmental conditions. Additionally, the database must be updated regularly to account for environmental changes, which introduces further complexity. Efficient data management strategies are essential to reduce the burden of data collection and maintenance. Sparse data collection methods can minimize the amount of data needed while still achieving high accuracy. Similarly, data augmentation techniques can reduce data collection efforts without compromising the quality of the fingerprints. By optimizing data management, fingerprinting-based systems can improve scalability and reduce the operational overhead associated with large-scale deployments.

1.2.7. EXPLAINABILITY AND TRANSPARENCY

While recent advances in deep learning have significantly improved the accuracy of fingerprinting-based localization systems, these models often operate as opaque black boxes, providing little insight into how decisions are made. This lack of interpretability hinders the trust,

accountability, and diagnosability of such systems—particularly in mission-critical applications like emergency response, indoor navigation for individuals with disabilities, and enterprise asset tracking. Without a clear understanding of what features influence the model’s predictions, it becomes difficult to debug localization errors, assess model fairness across environments, or verify its behavior under distributional shifts. To address these limitations, fingerprinting-based systems must incorporate mechanisms for explainable ML (XML). These include methods such as saliency mapping, feature attribution, and logic-based reasoning that allow users and developers to trace the influence of specific Wi-Fi access points or RSS values on the final location estimate. By exposing the internal reasoning of the model, XML not only facilitates transparency but also aids in identifying model weaknesses, detecting data biases, and ensuring regulatory compliance in sensitive domains.

1.3. DISSERTATION OVERVIEW

Indoor localization, though widely studied, remains challenging to deploy in practice due to the challenges outlined in Section 1.2. Addressing all of these challenges within a single, monolithic framework would introduce significant computational overhead. This makes real-time deployment on resource-constrained platforms like smartphones impractical, as it would degrade both battery life and system responsiveness—two critical factors for smartphone or mobile applications. To address this, the dissertation introduces *SPARK*, a modular and lightweight suite of ML frameworks specifically designed to tackle each challenge individually. *SPARK* leverages innovative, robust, and real-time-capable ML algorithms to ensure that indoor localization remains accurate, efficient, and deployable under practical conditions.

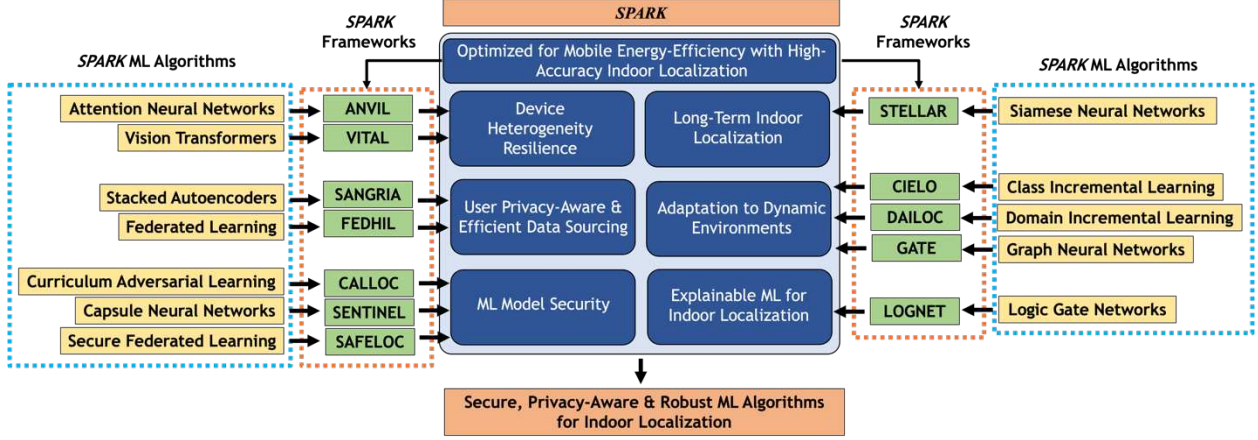


Figure 6: *SPARK* overview: challenges addressed (blue), corresponding *SPARK* frameworks (green), and underlying ML algorithms (yellow).

Figure 6 provides an overview of *SPARK*. The blue blocks represent the core challenges this dissertation addresses. The green blocks denote individual *SPARK* frameworks designed to overcome these challenges, while the yellow blocks highlight the novel ML algorithms introduced in this work. All *SPARK* components are developed with mobile deployment in mind, aiming for low inference latency, energy efficiency, and high localization accuracy on commercially available smartphones. The systems were tested using RSS fingerprint data collected from actual buildings, and all experiments were performed under realistic conditions—without introducing any artificial signal modifications—ensuring that evaluation results reflect true deployment scenarios.

To begin, this dissertation first investigates the effects of device heterogeneity on Wi-Fi fingerprinting systems, a key challenge in practical indoor localization. In Chapters 2 and 3, we analyze how different smartphone models perceive RSS signals differently, even under identical environmental conditions. This inconsistency severely degrades model performance when the training and inference devices differ. To mitigate this, we introduce ANVIL [45], [135] in Chapter 2, a novel attention-based neural network that improves generalization across device types while maintaining low computational cost. Chapter 3 extends this idea further with VITAL [80], [136],

which builds on vision transformer principles to offer enhanced heterogeneity resilience while remaining lightweight enough for real-time smartphone deployment.

Moving beyond device heterogeneity, Chapter 4 addresses long-term indoor localization under progressive environmental changes, such as furniture movement or human activity over weeks, months to several years. Here, we present STELLAR [134], a contrastive learning-based Siamese neural network that learns to recognize how RSS fingerprints evolve over time. This model is capable of sustaining localization performance without retraining, even after significant environmental changes.

The next challenge explored is efficient and privacy-aware data sourcing. Collecting labeled RSS fingerprints from multiple devices and locations is labor-intensive and raises privacy concerns. Chapter 5 introduces SANGRIA [82], a stacked autoencoder model that generates synthetic fingerprints reflecting device-specific noise characteristics. This drastically reduces the need for manual data collection from diverse hardware. Chapter 6 complements this by introducing FEDHIL [88], a federated learning (FL) framework that trains a central model without transferring user data. FEDHIL [88] also includes a novel mechanism to enhance privacy during training, ensuring secure model updates while maintaining performance across different devices.

Next, this dissertation turns to security and robustness under adversarial conditions. Indoor localization systems, like any other ML systems, are susceptible to crafted input manipulations caused from cyber or adversarial attacks that can cause prediction failures. In Chapter 7, we introduce CALLOC [87], a curriculum-based adversarial learning method that gradually exposes the model to increasingly complex attack patterns. This allows the model to build resilience incrementally. Chapter 8 builds on this by proposing SENTINEL [183], which uses a capsule neural network to isolate and suppress adversarial noise embedded in the RSS fingerprints,

achieving high accuracy even under strong attack scenarios. In Chapter 9, we consider adversarial threats under a federated learning setup. We present SAFELOC [259], a defense mechanism that uses saliency maps to filter out untrustworthy updates from clients during FL aggregation, preventing poisoned or unreliable model updates from degrading overall system performance.

Next, this dissertation addresses the need for adaptability in dynamic environments, where spatial layouts or RSS distributions may change over time. In Chapter 10, we propose CIELO [258], a class incremental learning model that allows the localization system to update itself over time while preserving previously learned information—overcoming the problem of catastrophic forgetting. Chapter 11 introduces DAILOC [260], a domain-incremental learning technique that enables the model to sequentially learn from new environments and devices, adapting to evolving domains without retraining from scratch. Chapter 12 adds spatial awareness through GATE [261], a graph neural network that learns the geometric structure of the indoor space, improving inference accuracy by encoding spatial relationships between the spatial layout of the indoor space and the corresponding RSS fingerprints.

Finally, Chapter 13 addresses the growing demand for explainability in ML (XML) systems. We introduce LogNet, [262] a logic gate-inspired interpretable model that reveals how traditional ML models make decisions under various conditions. LogNet [262] offers insights into which access points (APs) most influence predictions? how model behavior changes under noisy input? and how such knowledge can guide failure diagnosis or system improvement? This is a step toward making indoor localization not only robust but also transparent and trustworthy.

This dissertation concludes in Chapter 14 with a summary of contributions and suggestions for future research directions. Overall, *SPARK* is a modular and practical framework that advances

the field of indoor localization across robustness, scalability, privacy, and interpretability—
enhancing the field closer to real-world and intelligent indoor localization systems.

CHAPTER 2

ANVIL: SMARTPHONE INVARIANT INDOOR LOCALIZATION USING MULTI-HEADED ATTENTION NEURAL NETWORKS

Smartphones together with RSS fingerprinting serve as an efficient approach for delivering a low-cost and high-accuracy indoor localization solution. However, a few critical challenges have prevented the wide-spread proliferation of this technology in the public domain. One such critical challenge is device heterogeneity, i.e., the variation in the RSS signal characteristics captured across different smartphone devices. In the real-world, the smartphones or IoT devices used to capture RSS fingerprints typically vary across users of an indoor localization service. Conventional indoor localization solutions may not be able to cope with device-induced variations which can degrade their localization accuracy. We propose a multi-head attention neural network-based indoor localization framework that is resilient to device heterogeneity. An in-depth analysis of our proposed framework across a variety of indoor environments demonstrates up to 35% accuracy improvement compared to state-of-the-art indoor localization techniques.

2.1. INTRODUCTION

The proliferation of GPS (Global Positioning Systems) technology transformed the way we commute and engage with our surroundings. Now, GPS technology is a standard baked into every smartphone available on the market. This identical technology is now powering several geo-location-based businesses and other interactive platforms such as Uber, Pokémon-Go and location-based marketing. In the present times, indoor localization technology holds the innate potential to revolutionize navigation for indoor locales that are GPS deprived, e.g., malls, buildings, and

tunnels. Several firms such as IndoorAtlas, Target (Shopkick), and Zebra have already begun to provide services that aid customers in locating products within a store [1].

While GPS is a de-facto standard for outdoor localization, there are no similar global standards for indoor environments. As a result, a wide range of approaches have been created that employ a variety of sensors and radio frequencies. A few widely used radio signals for indoor localization include Bluetooth, RFID, UWB (Ultra-Wide Band), and Wi-Fi [2], [3], [4]. Due to its low setup costs and high ubiquity, Wi-Fi-based indoor localization has received the greatest academic attention [3]. This is further fueled by the fact that most people today own smartphones with Wi-Fi radios.

Despite the evident benefits of Wi-Fi based indoor localization, certain key challenges remain unresolved. Weak wall penetration, multipath fading, and shadowing effects all diminish the efficacy of such indoor localization platforms. Due to these issues, establishing a precise mathematical relationship between the Received Signal Strength Indicator (RSSI) and the relative distances from Wi-Fi Access Points (APs) is challenging. These issues serve as the motivation behind fingerprinting-based techniques that are independent of the RSSI-distance relationship, and therefore are resilient to aforementioned drawbacks of Wi-Fi-based indoor localization.

Traditionally, fingerprinting-based indoor localization comprises of two phases. In the first phase (offline or training phase), the provider of the localization service captures the RSS values for visible APs at various indoor locations of interest. This results in a database of RSS fingerprint vectors and associated locations or Reference Points (RPs). This database may further be used to train models (e.g., machine learning-based) for location estimation. In the second phase (online or testing phase), a user unaware of their own location captures an RSS fingerprint on a device such

as a smartphone. This fingerprint is then sent to the trained model to determine the user’s location. This location can be overlaid on a map and visualized on the smartphone’s display.

In the domain of fingerprinting-based indoor localization, a vast majority of work utilizes the same smartphone for (offline) data collection and (online) location prediction, e.g., [5], [6], [7]. This approach assumes that in a real-world setting, the localization devices operated by users would have identical signal characteristics. Such a premise is largely debunked by the variety of brands and models that make up the modern smartphone industry. In practice, the user base of smartphones is composed of heterogeneous device components and characteristics such as Wi-Fi chipset, antenna shape, OS version, etc. yielding variations in antenna gain. Recent works demonstrate that the observed RSS values for a given locale captured across diverse smartphones vary significantly [8]. This degrades the localization accuracy achieved through conventional fingerprinting and motivates smartphone heterogeneity or device invariant fingerprinting techniques.

2.2. CONTRIBUTIONS

In this chapter, we present a robust and computationally lightweight Wi-Fi RSSI-based fingerprinting framework (ANVIL) that aims to achieve device invariance such that it experiences minimal accuracy loss across heterogeneous smartphones. The main contributions of our work are:

- We conduct an analysis to highlight the impact of device heterogeneity on RSS fingerprints.
- Towards the goal of promoting generalized device invariance, we identify and adapt data augmentation methodologies for training deep machine learning models.

- We introduce a novel multi-head attention neural network for device invariant indoor localization.
- By collecting fingerprints with multiple heterogeneous devices across buildings, we create benchmarks to test the localization accuracy of ANVIL against the state-of-the-art.
- We prototype our calibration-free device invariant indoor localization framework, deploy it on smartphones, and evaluate it under real-world settings.

2.3. RELATED WORKS

A considerable amount of work has been dedicated to addressing the challenges associated with Wi-Fi fingerprinting based indoor localization. The recent growth in the computation capabilities of smartphones has enabled the proliferation of localization algorithms and frameworks with high computational and memory demands. For instance, Feed-Forward Deep Neural Networks (FF-DNN) [9] and also increasingly complex Convolutional Neural Networks (CNN) [6] combined with ensemble methods are being deployed on embedded devices to enhance indoor localization accuracy [7]. One cause for concern with the proliferation of such techniques are the acute energy limitations on smartphones. The work in [5] addresses this challenge in a limited manner. However, most prior works in the domain of fingerprinting-based indoor localization, including [5], are plagued by the same major drawback, i.e., the lack of an ability to adapt to device heterogeneity across the offline and online phases. This drawback, as discussed in Section 2.7, leads to unpredictable degradation in localization accuracy in the online phase.

In the offline phase, the localization service provider likely captures fingerprints using a device that is different from the IoT devices or smartphones employed in the online phase by users. Some common software and hardware differences that introduce device heterogeneity include Wi-Fi

antennas, smartphone design materials, hardware drivers, and the OS [4]. Techniques to overcome such issues fall into two major categories: calibration-based and calibration-free methods.

An indoor localization framework can be calibrated in either the offline or online phase. The work in [10] employs offline phase calibrated fingerprinting-based indoor localization. In this approach, the fingerprint collection process employs a diverse set of devices. Later, an autoencoder is specifically trained and calibrated using fingerprints from different devices. The role of the autoencoder is to create an encoded latent representation of the input fingerprint from one device such that decoded output fingerprint belongs to another device. In this manner, the latent representation created is expected to be device invariant. While this approach is promising, it comes with the significant overhead of utilizing multiple devices in the offline phase to capture fingerprints. Additionally, there are no guarantees that the set of devices employed to capture fingerprints in the offline phase capture ample heterogeneity that may be experienced by the localization framework in the online phase.

The online phase calibration approach involves acquiring RSS values and location data manually for each new device in the online phase [11]. Such an approach is however not very practical as it can be cumbersome for users. With this approach, once a user arrives at an indoor locale, they have to move to a known location and capture an RSS fingerprint. The RSS information is collected, and then manually calibrated through transformations such as weighted-least square optimizations and time-space sampling [12]. Crowdsourcing techniques may also be used in conjunction to strengthen these approaches. Unfortunately, such systems experience considerable accuracy degradation [13].

In calibration-free fingerprinting, the fingerprint data is frequently converted into a standardized format that is transferable across mobile devices. In an effort to achieve this

standardization, previous techniques such as Hyperbolic Location fingerprint (HLF) [14] and Signal Strength Difference (SSD) [15] employ ratios and differences between individual AP RSS values respectively. But these approaches suffer from low accuracy due to loss of critical distinguishing input fingerprint features in the transformation process. An alternative to the standardization of fingerprints is presented in [16]. The work proposed AdTrain, which employs adversarial training to improve the robustness of a deep-learning model against device heterogeneity. AdTrain introduces noise in the fingerprint and associated location label before training the deep-learning model. Based on our experiments, such an approach does improve the deep-learning model’s robustness to device heterogeneity in a limited manner. The approach used in our framework in this chapter (ANVIL) is orthogonal to the one proposed in AdTrain and ANVIL can be extended to include AdTrain. The work in [17] employs Stacked Auto-Encoders (SAE) for improving resilience to device heterogeneity. The authors expect that the lower dimensional encodings created using the SAE are more resilient to RSS variations across devices. However, based on our experimental evaluations (Section 2.7.3), we found that such an approach is unable to deliver high-quality localization and also does not converge easily. In contrast, ANVIL employs a multi-head attention neural network to achieve improvements in device invariance for indoor localization.

There are a limited set of previous works that have employed attention layers for magnetic [18], [19] and Channel State Information (CSI) fingerprint-based [21] indoor localization. However, these works do not consider device heterogeneity. Additionally, it is well known that magnetic fingerprints are unstable over time and heavily impacted by minor changes in the indoor environment. On the other hand, CSI enabled hardware is not available in off-the-shelf smartphones available today [20]. In contrast, ANVIL employs relatively stable Wi-Fi RSS that

can be captured through heterogeneous off-the-shelf smartphones of various vendors to deliver stable indoor localization.

Our proposed framework in this chapter, ANVIL, is a multi-head attention neural network based and calibration-free indoor localization framework that can be easily deployed to off-the-shelf smartphones, while relying on the ubiquity of Wi-Fi signals to deliver device invariant performance. To the best of our knowledge, no previous works have attempted to use attention in the context of improving device invariance. We performed extensive evaluation of ANVIL against state-of-the-art prior works and evaluated it over a variety of indoor environments and smartphones in real-world settings.

2.4. ANALYSIS OF HETEROGENEITY

To understand the cause of degradation in localization performance due to device heterogeneity, we evaluate the RSS fingerprints captured by two distinct smartphones. For this experiment, 10 fingerprints are captured at a single location with two smartphones: LG V20 (LG) and Oneplus 3 (OP3). The RSS values captured are in the range of -100dBm to 0dBm , where -100dBm indicates no received signal and 0dBm is the highest signal strength. The RSS values for the two devices are presented in Figure 7. The solid lines represent the mean values, whereas the shaded regions represent the range of the observed RSS values. From Figure 7, we can make the following key observations:

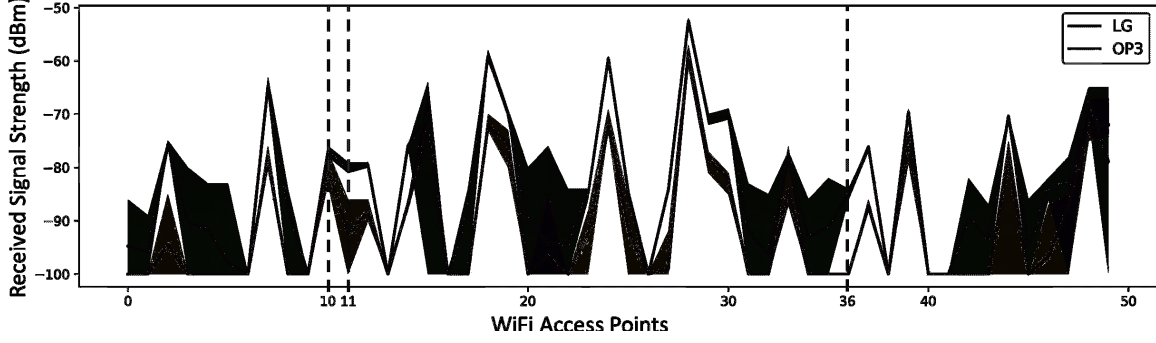


Figure 7: RSS values for Wi-Fi APs as simultaneously observed by two unique smartphones at a specific location. Solid lines represent the mean values. Shaded regions represent the range of RSS values, over 10 fingerprint readings. Distinct Wi-Fi APs (by MAC IDs) are denoted as unique integers from 0 to 50.

- There is considerable similarity in the shape of the RSS fingerprints across the two smartphones. Therefore, indoor localization frameworks that focus on pattern-matching-based approaches may be able to deliver higher quality localization performance.
- The RSS values for the LG device exhibit an upward shift (higher signal reception) by an almost constant amount. This constant shift of RSS value is similar to increasing the brightness of an image.
- We also observe a contrastive effect for certain parts of the fingerprints. By contrastive effect, we mean the difference between the RSS value changes. In Figure 7, we observe that while the RSS fingerprints across the two devices rise and fall together, the specific amount that they rise and fall by are not the same. A clear example of this observation can be seen across the RSS values of APs 10 and 11.
- Some APs that are visible when using the LG device are never visible (RSSI = -100dBm) to the OP3 device. A good example of this is AP 36, where the RSS value for the LG device is -80dBm and the RSS value for the OP3 device always remains -100dBm . From

the perspective of deep-learning models, this is similar to the random dropout of input AP RSSI.

- Lastly, we also observe that the RSS values from the LG device vary much more than the OP3 device, for a majority of the APs. This indicates that different smartphones may experience different amounts of variation in RSS values, and it may also be hard to quantify the range of RSS variation for a given device beforehand. Therefore, our proposed approach must be resilient to such unpredictable variations in the raw RSS values observed from an AP.

The evaluation of observed fingerprint variations across the two smartphones presented in this section (as well as our analysis with other smartphones that show a similar trend) guides the design of our device invariant ANVIL framework. We specifically base our data augmentation strategy in ANVIL on the observations made here. The multi head attention-based deep-learning approach (discussed in the next section) was chosen and adapted in ANVIL to promote generalized pattern matching across heterogeneous devices.

2.5. ATTENTION MECHANISM

The attention mechanism in the domain of deep learning is a relatively new approach initially employed for natural language processing (NLP) [22] and more recently also for image classification [23]. The concept of attention is derived from the idea of human cognitive attention and our ability to selectively focus on sub-components of information while ignoring less relevant components of the same. For the purpose of deep learning, the attention mechanism is modeled as the retrieval of attention information from a database containing key and value pairs. Given a query Q and a set of key-value pairs (K, V) , attention can be computed as shown in Equation 1:

$$Attention(Q, K, V) = similarity(Q, K)V \quad (1)$$

In the above expression, a variety of similarity functions may be employed such as dot-product, scaled dot product, additive dot product, etc. [18]. Among the variants, the dot-product attention is the fastest to compute and is the most space-efficient, making it our choice for this work. The process of computing scaled-dot-product attention (Figure 8; right) for a given set of queries, keys, and values (Q, K, V) can be captured in an equation as shown in Equation 2:

$$Attention(Q, K, V) = softmax\left(Q K^T / \sqrt{d_k}\right)V \quad (2)$$

where d_k is the dimensionality of the key vector. The role of the scaling factor $(\sqrt{d_k})$ is to counteract the effects of very large magnitudes being fed to the Softmax function, leading to regions that produce extremely small gradients. In practice, scaled dot-products are also known to outperform other approaches of computing attention.

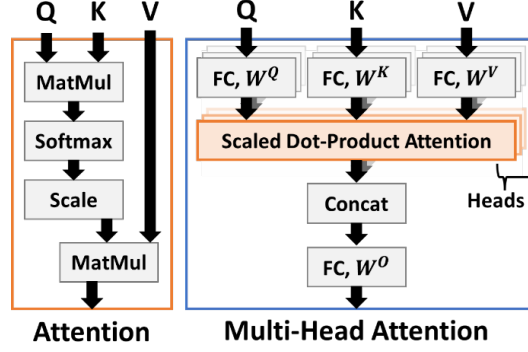


Figure 8: A procedural representation of scaled-dot-product attention (left) and multi-head attention (right). Each attention layer accepts three inputs: Query (Q), Keys (K), and Values (V).

While attention serves as a low-overhead approach for capturing a weighted relationship between queries and values, given a set of key-value pairs, it lacks any learnable parameters. This limits its ability to identify and quantify the hidden relationships between the different pairings of (Q, K, V) . The work in [22] extends the idea of a singular attention computation with multiple distinct versions (multiple heads) of linearly projected queries, keys, and values. Each of these learnable linear projections are of dimension d_k , d_q , and d_v . This form of attention is more commonly known as multi-headed attention and can be formally captured by the following Equation 3:

$$MultiHead(Q, K, V) = Concat(h_1, h_2, h_3 \dots h_n)W^O$$

$$where h_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

where W_i^Q , W_i^K , W_i^V and W_i^O are model parameters or weights associated with the linear projections with dimensions $head\ size \times d$. Head size (HS) is a hyperparameter of the multi-head attention layer and d is the length of vector being projected (query, key or value). The process of

computing scaled-dot-product attention and multi-headed attention is depicted in Figure 8. The computation associated with each head can be performed in parallel, as shown.

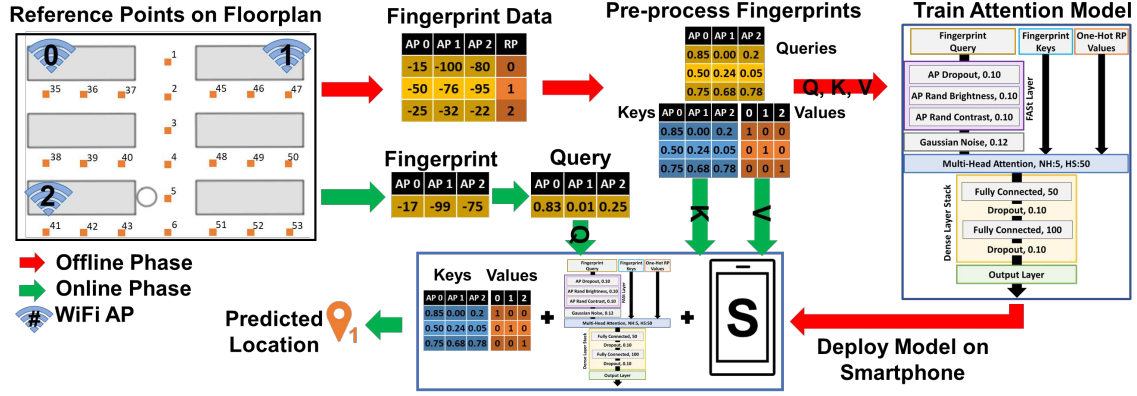


Figure 9: An overview of the ANVIL indoor localization framework depicting the offline (red arrows) and online (green arrows) phases.

2.6. ANVIL FRAMEWORK

2.6.1. FRAMEWORK OVERVIEW

Figure 9 presents a high-level representation of our proposed *ANVIL* framework. We begin in the offline phase that is annotated by red arrows. Here we capture RSS fingerprints for various RPs (see Section 2.7 for details) across the floorplan of the building. Each row within the RSS database consists of the RSS values for every AP visible across the floorplan and its associated RP. The RSS values indicated in a given row were captured using a single Wi-Fi scan. These fingerprints are then pre-processed into queries (Q), keys (K) and values (V) to train a multi-headed attention neural network model, as shown in Figure 9. The specific details of fingerprint preprocessing, model design and training are covered later in this section. Once the model has been trained, it is deployed on a smartphone, with the fingerprint key-value pairings. This concludes the offline stage.

In the online phase (green arrows), the user captures an RSS fingerprint vector at an RP that is unknown. For any Wi-Fi AP that was visible in the offline phase and is not observed in this online phase, its RSS value is assumed to be -100dBm , ensuring consistent RSS vector lengths across the phases. This fingerprint is pre-processed (see Section 2.6.2) to form the fingerprint query and sent to the deployed multi-head attention neural network model on the smartphone. As shown in Figure 9, the model when fed the online phase fingerprint query, along with the keys and values from the offline phase, produces the location of the user in the online phase. This location is then shown to the user on the smartphone’s display.

2.6.2. RSS FINGERPRINT PRE-PROCESSING

The RSS for various Wi-Fi APs along with their corresponding reference points are captured within a database as shown in Figure 9. As mentioned earlier, the RSS values vary in the range of -100dBm to 0dBm , where -100 indicates no signal and 0 indicates a full (strongest) signal. The RSS values captured in this dataset are normalized to a range of 0 to 1 , where 0 represents the weak or null signal, and 1 represents the strongest signal. This new dataset is the basis of all training data required by our multi-head attention model. As discussed in Section 2.5, the multi-head attention model requires three main inputs: the Queries and the Key-Value pair dataset. For this work, the RSS fingerprint vectors captured in the training phase (without RP information) are used as both queries and keys in the offline phase. The associated RPs are one-hot encoded and are used as values.

2.6.3. FINGERPRINT AUGMENTATION (FAST) STACK

A major challenge to maintaining localization stability for fingerprinting-based indoor localization is the variation in RSS fingerprints across heterogeneous devices, as discussed in Section 2.4. However, in the online phase, it would be impossible to foretell what combination of effects the fingerprint captured using a smartphone. The received RSS fingerprints can vary depending on external factors like building layout, walls, metallic object (reflecting the RSS fingerprint) making the prediction of the location ambiguous.

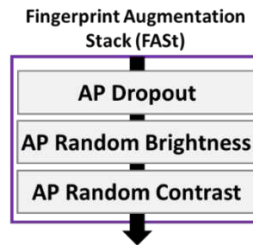


Figure 10: Various components of the Fingerprint Augmentation Stack (FAST). Each subcomponent augments a specific heterogeneity effect.

To overcome this challenge, we propose a streamlined Fingerprint Augmentation Stack (FAST) implemented as a layer that contains the required subcomponents for the augmentation of RSS fingerprints that would promote the resilience to device heterogeneity. The major advantage of FAST is that it can be seamlessly integrated into any deep-learning based model and promote the rapid prototyping of device invariant models for the purpose of fingerprinting-based indoor localization. A procedural representation of the FAST layer is shown in Figure 10.

The design of the FAST layer is based on the three observations from our analysis in Section 2.4, i.e., AP dropout, contrast, and brightness. While these are well known data augmentation techniques in the domain of computer vision [24], they do not directly translate to the domain of

RSS fingerprinting-based pattern matching. In the computer vision domain, all three effects are applied to the inputs indiscriminately, as it is most likely that the input does not contain cropped portions relative to rest of the image (blackened/whited out regions of image). This is especially true in the domain of image-based pattern matching, where the input image may not have cropped out components, with the exception of image inpainting [25]. The utilization of image augmentation before training deep-learning models is a well-known approach for improving generalizability.

In the case of RSSI-fingerprints, the fact that a certain set of APs are not visible in some areas of the floorplan as compared to the others is a critical and unique attribute. A deep-learning framework might utilize this to better correlate an RSS pattern to a specific location on the floorplan. Based on these beliefs and our analysis in Section 2.4, we adapt the general random dropout, brightness, and contrast layers to only act on Wi-Fi APs that are visible to the smartphone i.e., $RSSI \neq -100 \text{ dB}$. It is important to note that we deliberately, do not add random noise to the FAST layer, as it was not specially adapted for the purpose of RSS fingerprinting-based indoor localization. However, random noise is an important aspect of the model design as discussed in the next subsection.

2.6.4. MULTI-HEADED ATTENTION MODEL

The concept of attention and especially multi-head has gained considerable popularity in various domains of pattern matching [22]. Its simplicity in mathematical operations enables a computationally efficient process for deep learning. It is also well known that conventional feed-forward neural network layers (dense layers) require lower computational capabilities on the target deployment platform when compared to convolutional approaches. Additionally, recent work in the domain of pattern matching is slowly considering attention as an alternative to convolutional

layers [26]. Based on these observations, we combine our domain specific fingerprint augmentation layer stack (FAST), Multi-Head Attention, and feed-forward neural networks to create a deep-learning classification model as depicted in Figure 11.

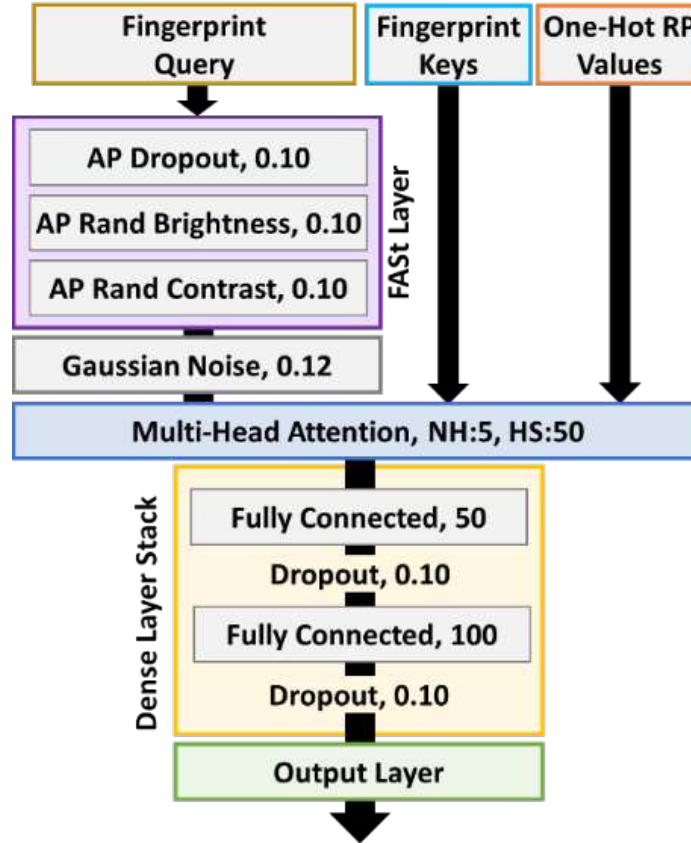


Figure 11: Overview of the multi-head attention neural network model in the *ANVIL* framework.

The model shown in Figure 11 consists of three inputs: fingerprint query, keys, and one-hot encoded reference point (RP) labels as values. The key-value inputs remain fixed in the offline and online phases of the model development process. The fingerprint keys and RP values are the fingerprints captured at specific RPs using the smartphone from the offline phase. In the offline phase, we use the same fingerprints as queries to train the model, whereas, in the online phase,

fingerprints captured by the end-user are fed as queries to the model which in turn produces the location of the user.

For the sake of simplicity, all hyperparameters in the model were chosen such that the model generalizes well across all the floorplans discussed in the experiments section. We employ the value of 0.10 for the dropout, random brightness, and contrast functions of the FAS_t layer. Therefore, for each RSS fingerprint Query fed to the model, random dropout, random contrast, and random brightness (increase or decrease) are applied with a 10% probability. Similarly, the Gaussian noise layer was set to a standard deviation of 0.12. No augmentation is applied to the fingerprint keys. Based on our experiments (not presented for brevity), the overall performance of the model is better when Gaussian noise is applied to the whole fingerprint, instead of the masked approach ($RSSI \neq -100$ dBm) used for the FAS_t layer. One possible explanation is that the noise layer acts as regularization on the visible AP dropout layer which only acts on visible APs. Following the augmentation of queries, they are then fed to the multi-head attention layer which has a total of 5 heads (NH) and a head size of 50 (HS). For simplicity of model design, the head size is used as the size of all linear projections within the multi-head attention layer. The output from the multi-head attention layer is fed to a stack of feed-forward fully connected or dense neural network layers with an interleaved dropout of 0.10. We employ the *ReLU* activation function across all layers, except for the output layer which uses *Softmax*. The length of the output layer is set to the number of unique RPs for the given floorplan. With this setup, our multi-head attention model design has approximately 111K trainable parameters.

2.7. EXPERIMENTS

2.7.1. EXPERIMENTAL SETUP

We evaluate the device invariance of *ANVIL* against four fingerprinting-based indoor localization frameworks from prior work across a benchmark suite containing five indoor paths in different buildings around the Colorado State University campus. Figure 12 depicts the floorplans, with each fingerprinted location or reference point (RP) denoted by a yellow dot. We chose a granularity of 1-meter for our experiments (distance between each yellow dot) which we believe is sufficient for the purpose of localizing humans.

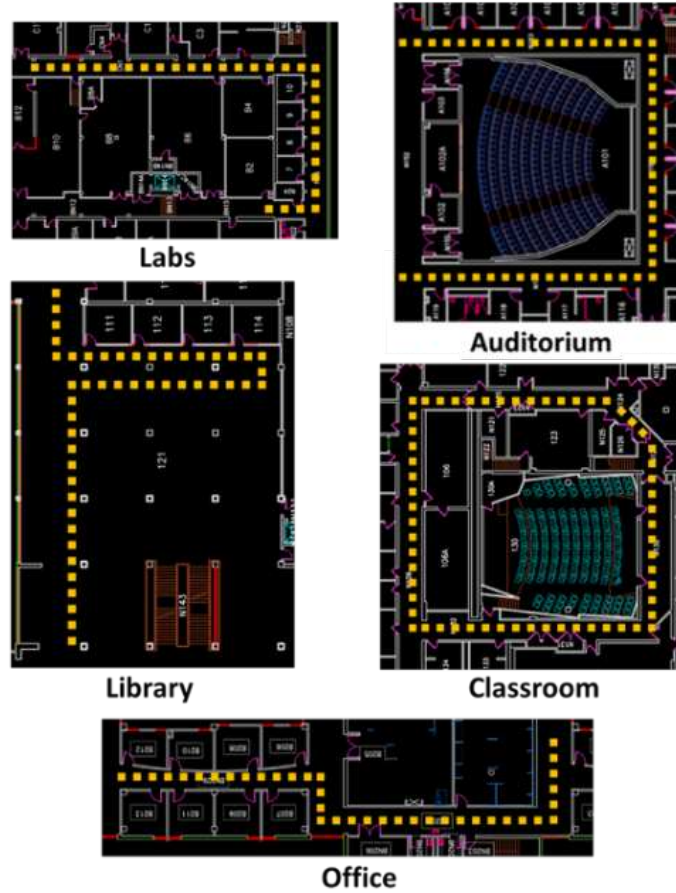


Figure 12: Indoor floorplans within the benchmark suite. Reference points are annotated with yellow boxes.

Each path (ranging 60 to 80 meters) was selected due to its salient features that may impact indoor localization. The *Classroom* floorplan is part of one of the oldest buildings on campus that is constructed from wood and concrete. This path is surrounded by a combination of labs that hold heavy metallic equipment as well as large classrooms with open areas. A total of 81 unique Wi-Fi APs were visible on this path. The *Auditorium* and *Library* floorplans are part of relatively new buildings on campus that have a mix of metal and wooden structures with open study areas and bookshelves. We observed 130 and 300 unique APs on the Auditorium and Library floorplans, respectively. The experiments performed do not consider factors such as the physical placement of the APs, the channel each AP is set to (assumed to be automatic), and the frequency at which an AP is broadcasting (2.5 or 5 GHz). This is a deliberate choice made to highlight the advantages for using fingerprinting-based indoor localization.

Table 1: Details of smartphones used in experiments.

Smartphone	Chipset	Android version
OnePlus 3 (OP3)	Snapdragon 820	8.0
LG V20 (LG)	Snapdragon 820	7.0
Moto Z2 (MOTO)	Snapdragon 835	8.0
Samsung S7 (SS7)	Snapdragon 820	7.0
HTC U11 (HTC)	Snapdragon 635	8.0
BLU Vivo 8 (BLU)	MediaTech Helio P10	7.0

The *Office* path is on the second floor of an Engineering building that is surrounded by small offices and covered by 180 APs overall. The *Labs* path is in the Engineering basement and is surrounded by labs consisting of a sizable amount of electronic and mechanical equipment with about 120 visible APs. Large quantities of metal and electronics on the Office and Labs paths lead to noisy Wi-Fi fingerprints that hinder indoor localization efforts. The fingerprints at both locations were captured in usual working hours and so the human occupancy of these indoor locales is not

artificially manipulated in any way. A total of 8 fingerprints per RP are used for training *ANVIL* in the offline phase and 2 fingerprints per RP are used in the online phase. We employ six smartphones from distinct vendors to capture the Wi-Fi fingerprints across the five floorplans shown in Figure 12. The specifications of these smartphones are listed under Table 1.

2.7.2. COMPARISION WITH PRIOR WORKS

Four state-of-the-art were implemented to establish the efficacy of our proposed *ANVIL* framework. The first work (*LearnLoc* [5]), builds on the K-Nearest-Neighbor (KNN) using a lightweight Euclidean distance-based metric to match fingerprints. Interestingly, the authors of [27] performed an error analysis to understand the merits of distance calculations. The primary focus was to create a comparison of Euclidean versus Manhattan distance-based metrics to match fingerprints. The work shows evidence of Euclidean distance-based metric to match fingerprints being more accurate. These works are incognizant of device heterogeneity and thus, *LearnLoc* [5] serves as a one of our motivations. The second work, *AdTrain* [17], is a deep learning-based approach that achieves device invariance through the addition of noise at the input and output labels. This creates an adversarial training scenario where the feed-forward neural network-based model attempts to converge in the presence of high-noise (adversity). The third work, *SHERPA* [28], is similar to the KNN-based approach in [2], [5] but is enhanced to withstand variations across devices in the offline and online phases of fingerprinting-based indoor localization. *SHERPA* achieves variation resilience by employing Pearson’s correlation as a distance metric to match the overall pattern of the fingerprints instead of a Euclidian distance-based approach in [5]. Lastly, the fourth work employs Stacked Auto-Encoders (*SAEs*) [16] designed to sustain stable localization accuracy in the presence of device heterogeneity across the offline and online phases.

The authors of *SAE* propose using an ensemble of stacked autoencoders (denoising autoencoders) to overcome the variation in RSS fingerprinting across different devices. A Gaussian process classifier with a radial basis kernel is then employed to produce the final location of the user.

		LearnLoc					SHERPA					AdTrain					SAE					ANVIL									
Offline Devices	S7	0.48	4.71	1.63	1.80	2.08	1.88	0.50	4.00	1.60	1.66	1.81	1.82	1.34	3.75	1.59	1.85	2.31	2.09	2.93	7.18	4.79	3.70	5.58	4.69	1.21	2.65	1.45	1.82	1.83	1.91
	BLU	2.62	1.63	2.60	2.83	1.79	2.67	2.74	1.79	2.76	2.76	1.84	2.53	3.27	1.57	2.46	2.68	2.34	2.91	5.26	4.67	5.84	5.83	5.95	6.27	2.77	1.89	2.52	2.61	2.30	2.99
	HTC	1.84	5.96	0.00	1.69	1.95	2.01	2.07	5.42	0.00	1.57	1.97	2.24	1.71	3.40	0.57	1.47	2.14	1.80	2.70	5.11	1.84	2.76	3.09	2.83	1.72	4.26	0.68	1.44	1.77	1.90
	LG	1.92	5.09	1.53	0.49	2.29	3.05	2.39	5.23	1.89	0.50	2.08	2.12	2.09	3.83	1.46	1.08	2.62	2.56	3.51	5.62	2.38	2.02	4.46	4.16	1.98	3.88	1.20	0.88	2.19	2.08
	MOTO	1.74	3.27	1.53	1.70	0.06	1.79	1.57	3.28	1.81	1.85	0.07	1.83	1.96	3.09	1.61	1.86	1.01	1.82	3.40	5.39	3.98	3.70	2.23	2.98	1.79	2.85	1.84	1.61	0.88	1.73
	OP3	1.56	4.58	1.69	1.62	1.59	0.57	1.77	4.29	1.78	1.60	1.85	0.55	1.92	3.03	1.76	2.05	1.62	1.13	5.33	3.85	5.88	5.40	2.65	1.95	1.58	3.01	1.39	1.78	1.67	1.29
	S7	0.58	1.72	1.25	1.57	1.47	1.69	0.60	1.83	1.14	1.50	1.43	1.62	1.10	1.86	1.44	1.67	1.33	1.83	1.87	2.49	2.41	2.21	2.58	2.61	0.85	1.53	1.16	1.41	1.40	1.49
	BLU	1.50	0.64	1.59	1.53	1.62	1.82	1.72	0.72	1.72	1.66	1.78	1.86	1.39	1.18	1.77	1.84	1.95	1.76	2.30	2.13	2.72	2.43	2.88	2.85	1.32	0.94	1.25	1.70	1.37	1.44
	HTC	1.37	2.02	0.03	1.61	1.63	1.94	1.36	1.95	0.03	1.63	1.78	1.85	1.46	1.98	0.75	1.71	1.81	2.03	1.92	2.25	1.44	2.10	2.50	2.65	1.12	1.46	0.36	1.36	1.43	1.70
	LG	1.81	2.23	1.65	0.44	1.86	2.75	1.57	2.03	1.52	0.43	1.73	1.87	1.64	1.90	1.25	0.95	1.42	2.00	2.41	3.42	2.35	1.77	2.86	3.39	1.23	1.54	0.99	0.68	1.38	1.72
	MOTO	1.65	1.65	1.61	1.72	0.18	1.97	1.71	1.77	1.74	1.74	0.21	1.99	1.56	2.06	1.61	1.69	0.89	1.91	2.21	2.36	2.35	2.43	1.97	2.85	1.51	1.58	1.55	1.66	0.52	1.86
	OP3	1.41	1.66	1.47	2.03	1.81	0.79	1.40	1.47	1.36	1.59	1.74	0.74	1.73	1.59	1.99	1.88	2.09	1.15	2.38	2.34	2.27	2.29	2.77	2.27	1.46	1.43	1.51	1.89	1.57	1.01
	S7	0.16	1.61	1.64	1.52	2.39	1.68	0.28	1.82	1.93	1.48	2.24	1.53	0.72	1.30	1.54	1.80	1.84	1.44	4.41	4.35	4.38	6.60	5.17	4.47	0.64	1.08	1.36	1.07	1.70	0.97
	BLU	1.46	0.39	1.79	1.48	2.90	1.55	1.36	0.52	1.80	1.54	2.33	1.84	1.75	1.13	1.67	1.66	1.83	1.52	4.41	3.91	3.66	5.02	5.69	5.17	1.21	0.80	1.49	1.38	1.59	1.47
HTC	1.30	1.69	0.03	1.30	2.60	1.96	1.25	1.80	0.03	1.38	2.00	1.71	1.51	1.30	0.48	1.54	2.06	1.88	3.69	3.99	3.04	3.85	4.77	4.32	1.02	1.21	0.28	1.22	1.62	1.52	
LG	1.84	3.32	1.55	0.54	3.02	3.0	1.61	2.34	1.67	0.61	2.30	1.99	1.67	2.15	1.56	1.00	2.43	2.20	3.64	4.04	3.04	3.50	6.03	5.41	1.18	1.69	1.16	0.43	1.60	1.60	
MOTO	1.48	2.24	1.74	1.65	1.38	1.20	1.38	2.47	1.54	1.29	1.31	1.74	1.84	1.75	2.45	1.87	2.15	1.79	6.14	4.40	8.03	6.55	4.07	3.64	1.33	1.36	1.39	1.33	1.13	1.08	
OP3	1.33	1.89	1.68	1.61	2.03	0.36	1.39	1.95	1.65	1.74	2.14	0.38	1.51	1.73	1.76	1.91	1.84	1.18	5.07	3.84	6.03	5.86	4.01	3.59	1.13	1.28	1.66	1.34	1.59	1.20	
S7	0.63	2.53	1.66	2.85	2.47	2.41	0.73	2.96	1.97	3.08	2.07	2.46	0.56	2.90	1.43	2.29	1.33	2.12	3.17	3.59	3.18	3.30	3.91	4.43	1.04	2.04	1.34	1.92	1.62	1.79	
BLU	2.12	0.83	2.34	3.84	2.17	2.88	2.89	0.77	2.72	3.83	2.82	3.48	2.94	0.71	2.56	3.60	2.76	3.06	4.22	4.03	3.13	3.70	5.38	6.93	2.63	1.28	2.74	4.13	2.31	2.63	
HTC	1.67	3.72	0.00	2.19	2.88	3.06	1.70	3.03	0.00	2.49	1.72	2.51	1.85	2.39	0.10	3.24	1.90	3.05	4.95	5.77	3.19	3.57	5.42	5.73	1.37	2.40	0.20	2.35	1.82	2.73	
LG	3.36	4.37	2.22	0.13	4.27	4.63	2.49	3.18	2.02	0.17	2.34	3.34	2.31	4.36	2.12	0.12	2.38	3.65	4.99	5.44	3.45	2.71	5.56	5.67	2.53	3.12	2.50	0.63	2.01	3.39	
MOTO	1.35	1.98	1.44	2.67	0.03	2.53	1.39	2.04	1.50	2.15	0.03	2.32	1.58	2.49	1.52	2.19	1.82	2.61	4.21	3.99	4.24	3.43	4.12	4.47	1.47	2.14	1.10	1.84	0.85	2.29	
OP3	2.68	3.23	2.52	3.64	2.71	0.25	2.72	3.56	2.68	3.90	2.91	0.29	2.06	3.75	2.40	3.13	1.68	0.55	5.00	5.99	6.28	6.41	4.90	3.75	1.98	3.21	2.00	2.99	2.08	0.80	
S7	0.08	1.87	1.32	1.46	2.52	1.70	0.08	1.74	1.36	1.56	1.93	1.70	0.62	1.22	1.19	1.55	1.82	1.32	2.31	2.87	2.12	2.09	3.34	3.14	0.60	1.15	1.33	1.18	1.69	1.08	
BLU	1.46	0.50	1.78	1.94	1.76	2.06	1.48	0.40	1.54	1.52	1.68	1.85	1.33	1.14	1.72	1.56	2.03	1.77	2.71	3.16	2.45	2.32	4.31	3.68	1.09	0.62	1.56	1.23	1.56	1.51	
HTC	1.07	2.13	0.04	1.39	2.25	1.74	1.05	1.59	0.04	1.35	1.63	1.39	1.22	1.52	0.44	1.48	2.29	1.87	3.22	3.30	2.77	2.84	3.59	3.37	1.09	1.10	0.25	1.37	1.55	1.11	
LG	1.85	2.18	1.66	0.23	3.15	2.72	1.58	2.18	1.68	0.23	2.19	1.76	1.15	1.44	1.38	0.50	2.23	1.95	2.92	3.36	2.29	1.74	3.66	3.85	1.02	1.31	1.23	0.50	1.78	1.37	
MOTO	1.39	1.98	1.53	1.89	1.23	1.59	1.49	2.11	1.49	1.68	1.77	1.55	2.00	1.82	2.15	2.85	1.02	1.57	2.25	2.64	1.69	2.07	2.51	2.51	1.17	1.43	1.24	1.39	1.12	1.25	
OP3	1.08	2.27	1.55	1.90	1.44	0.25	1.11	2.23	1.31	1.27	1.78	0.21	1.48	1.61	1.72	2.17	1.35	0.85	2.18	2.37	1.83	2.29	2.58	2.21	0.96	1.26	1.04	1.81	1.19	0.59	
		S7	BLU	HTC	LG	MOTO	OP3	S7	BLU	HTC	LG	MOTO	OP3	S7	BLU	HTC	LG	MOTO	OP3	S7	BLU	HTC	LG	MOTO	OP3	S7	BLU	HTC	LG	MOTO	OP3
		Online Devices																													

Figure 13: The mean localization error in meters for the framework ANVIL proposed in [22] as compared to state-of-art previous works.

2.7.3. EXPERIMENTAL RESULTS

A color-coded tabular compilation of the mean localization accuracy in meters across all floorplans in the benchmark suite using every combination of offline (training) and online (testing) devices is presented in Figure 13 Each row-group in Figure 13 captures the mean indoor localization error in meters for a single floorplan. The device abbreviations on the vertical axes

indicate the smartphones used in the offline phase, whereas the devices listed on the horizontal axes indicate smartphones employed by a user in the online phase. The observed localization errors are color coded from green (lower/better) to red (higher/worse) per floorplan.

From Figure 13, we immediately observe that the general performance of all five localization frameworks varies across various floorplans. The possible explanation for such an observation could be the differences in path length, overall visibility of Wi-Fi APs, path shape, and other environmental factors. The variances of results across different floorplans also highlights the importance of considering a diverse set of environments when evaluating fingerprinting-based indoor localization frameworks. When making observations about the performance of the different localization frameworks we note that across the whole benchmark suite, *LearnLoc* (with the exception of *SAE*, discussed later) delivers the least stability in the presence of device heterogeneity. The simple introduction of Pearson correlation-based pattern-matching metric through *SHERPA* greatly improves the overall localization accuracy. However, there are some offline-online combinations of smartphones where *SHERPA* is not very effective. For example, in the case when the LG-BLU (offline-online) is used in the Auditorium floorplan, *SHERPA* is not very effective. Based on our empirical analysis, the BLU device exhibits high noise across fingerprints of the same location. Traditional machine learning models, unlike their deep-learning counterparts, are unable to withstand such noisy fingerprints. In contrast, *AdTrain* and *ANVIL* are relatively resilient to such noisy combinations of devices which can be attributed to the benefits from adversarial training and FASt layer for fingerprint augmentation, respectively.

From Figure 13, we also note that *SAE* produces stable localization performance across device combinations i.e., different offline-online smartphone combinations produce similar localization error. However, the localization errors by themselves are extremely large. Due to such poor

performance of *SAE*, we no longer consider it in the subsequent experiments in this chapter. *AdTrain* shows the best resilience to device heterogeneity second only to our *ANVIL* framework. We suspect that the simplistic input and label noise induction plays a critical role in the success of *AdTrain*. An interesting observation is that the approaches proposed by the authors in *AdTrain* appear to be orthogonal to the ones proposed by us for the *ANVIL* framework. While *ANVIL* outperforms *AdTrain*, its performance could be further improved by combining the approaches of both frameworks.

Even though Figure 13 presents a detailed view of the pair-wise evaluation of various offline-online devices, it is difficult to derive generalized conclusions for the performance of individual frameworks. We therefore capture the average localization errors and associated standard deviations of all offline-online device pairs in Figure 14. From Figure 14, we note that as compared to *LearnLoc*, both *AdTrain* and *SHERPA* produce better results, with the exception of *AdTrain* in the Classroom and Labs floorplan. In contrast, *ANVIL* consistently produces better results across all floorplans and is more stable (lower standard deviation). *ANVIL* is able to perform up to ~30% better as compared to *SHERPA* and up to ~35% better than *Adtrain* on the Labs and the Office paths.

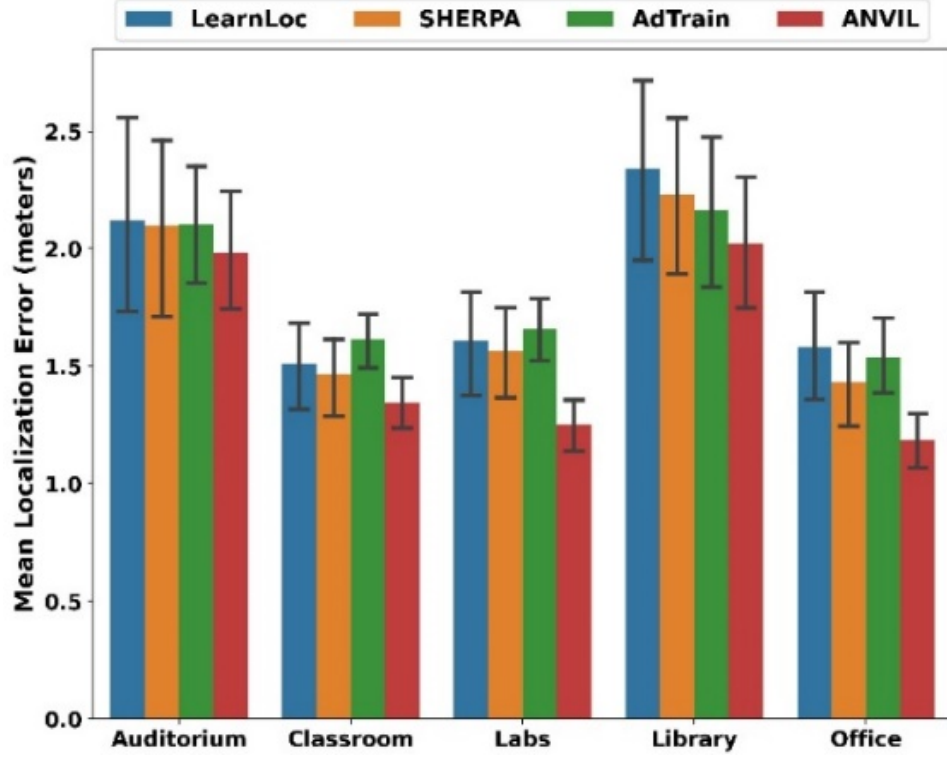


Figure 14: The localization performance of *ANVIL* as compared to state-of-the-art previous works.

In general, based on our experimental analysis in this subsection, we conclude that our proposed *ANVIL* framework outperforms all previous state-of-the-art frameworks considered in this chapter, and delivers superior localization accuracy (low error) and stability (similar error across smartphone pairs via low standard deviation) across a diverse set of smartphones and benchmark floorplans.

2.8. CONCLUSION

In this chapter, we presented a novel framework called *ANVIL* that combines smart fingerprint augmentation as a layer together with multi-head attention in a neural network, for device invariant indoor localization. We evaluated Wi-Fi RSS fingerprints from smartphones of different vendors

and made key empirical observations that informed our fingerprint augmentation strategy. Our proposed framework was evaluated against several contemporary indoor localization frameworks using six different smartphones across five diverse indoor environments. Through the evaluations, we deduced that *ANVIL* delivers considerable resiliency to device heterogeneity and provides up to 35% better performance compared to the previous works. Our ongoing work is focusing on exploring incorporating secure indoor localization techniques [29], efficient model compression [30], [31], deep learning model engineering [6], and long-term fingerprint aging resilience [32] into *ANVIL*.

CHAPTER 3

VITAL: HETEROGENEITY RESILIENT INDOOR LOCALIZATION USING VISION TRANSFORMER NEURAL NETWORKS

Wi-Fi fingerprinting-based indoor localization is an emerging embedded application domain that leverages existing Wi-Fi access points (APs) in buildings to localize users with smartphones. Unfortunately, it has been demonstrated that the heterogeneity of wireless transceivers among various cellphones used by consumers reduces the accuracy and dependability of localization algorithms. In this chapter, we propose a novel framework based on vision transformer neural networks called VITAL that addresses this important challenge. Experiments indicate that VITAL can reduce the uncertainty created by smartphone heterogeneity while improving localization accuracy from 41% to 68% over the best-known prior works. We also demonstrate the generalizability of our approach and propose a data augmentation technique that can be integrated into most deep learning-based localization frameworks to improve accuracy.

3.1. INTRODUCTION

Indoor localization systems (ILS) allow the localization of items or persons within buildings and subterranean facilities (e.g., underground tunnels, mines). As the Global Positioning System (GPS) is unreliable in indoor settings due to the lack of visible contact with GPS satellites, an ILS must rely on alternative means of localization. These include techniques such as triangulation and fingerprinting with wireless technologies, e.g., Wi-Fi, Bluetooth Low Energy (BLE), Ultra-Wideband (UWB), and Radio Frequency Identification (RFID) [2]. Such ILS techniques are now powering several geo-location-based embedded systems and business platforms including

Amazon warehouses robots, parking features in self-driving cars, IoT indoor navigation solutions, and so on [33].

Out of the available approaches, fingerprinting has shown to be one of the most scalable, low-cost, and accurate indoor localization solutions [34], [3]. The operation of fingerprint-based ILS is divided into two phases. The supplier of the ILS gathers received signal strength indication (RSSI) data for wireless Access Points (APs), such as Wi-Fi APs, at various indoor locations of interest during the first (offline) phase. The unique RSS values at each location form a “fingerprint” vector at that location, which is different from the fingerprint vector at other locations. Subsequently, a database of RSS fingerprint vectors and their corresponding locations is created. This database can be used together with patterns matching algorithms to estimate location in the second (online) phase, where a person uninformed of their location captures an RSS fingerprint using their smartphone and sends it to the algorithm (on the smartphone or a cloud server) to predict their location. This predicted location can be displayed on the smartphone for real-time location visualization.

In recent years, Wi-Fi-based fingerprinting when combined with machine learning has been shown to provide promising accuracy for indoor location predictions [4]. Neural networks can capture prominent features in RSS values from Wi-Fi signals captured by a smartphone and map these features to the location of the smartphone, which is often referred to as the Reference Point (RP) prediction problem. Despite the claimed benefits of Wi-Fi fingerprinting-based indoor localization, there remain several unresolved issues. Wi-Fi signals are affected by poor wall penetration, multipath fading, and shadowing. Due to these difficulties, establishing a deterministic mathematical relationship between the RSS and distance from APs is challenging, especially in dynamic environments and across different device configurations. The latter problem is

particularly challenging as different smartphones use different wireless transceivers, which significantly changes RSS values captured by the different smartphones at the same location.

Today's indoor localization solutions are highly vulnerable to heterogeneous variations within smartphones. The performance of indoor localization solutions are susceptible to heterogeneity within smartphones, as well as other embedded and IoT devices that may participate in ILS. We quantify the impact of this device heterogeneity on several smartphone devices. This device heterogeneity leads to unpredictable variation in performance (location estimates) across users that may be present at the same physical location. The variations reduce precision, and ultimately the accuracy of ILS.

3.2. CONTRIBUTIONS

In this chapter, we present VITAL, a novel embedded software framework that enables robust and calibration-free Wi-Fi-based fingerprinting for smartphones, with broad applicability to embedded and IoT devices used in ILS. VITAL aims to achieve device invariance with the end goal of achieving negligible precision error across heterogeneous smartphones and high-accuracy localization. Our framework employs vision transformer neural networks for the first time to the problem of Wi-Fi fingerprinting-based indoor localization. RSS data is converted to images and processed using novel algorithms that integrate the transformer neural network to assist with device heterogeneity-resilient indoor localization. The main contributions of our work are:

- We present a novel vision transformer neural network-based indoor localization solution that is resilient to device heterogeneity and achieves high-accuracy localization.
- We propose a novel RSS image-based model to capture prominent features from Wi-Fi RSS fingerprints.

- We design a data augmentation technique that can be seamlessly integrated into any deep learning model, including vision transformers, to improve heterogeneity resilience.
- We evaluate the localization performance of VITAL against the best-known indoor localization solutions from prior work, across several real indoor paths in multiple buildings, and for various smartphones carried by users.

3.3. RELATED WORK

Indoor localization competitions in recent years held by Microsoft [35] and NIST [36] have advocated for Wi-Fi fingerprinting-based indoor localization as the foundation for future ILS. The improvements in the computational abilities of smartphones and embedded platforms have empowered them to execute deeper and more powerful machine learning (ML) models to improve indoor localization performance. Thus, many efforts in recent years have started to explore the use of ML with Wi-Fi fingerprinting.

A survey of ML-based Wi-Fi RSS fingerprinting schemes, including data preprocessing and ML prediction models for indoor localization, was presented in [37]. In [38], an analysis was performed to determine how to select the best RSS fingerprints for ML-based indoor localization. In [39] and [6], Deep feedforward Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) were shown to improve indoor localization accuracies over classical localization algorithms. A recent work in [40] explored the traditional transformer neural network on Channel State Information (CSI) data for indoor localization. The work in [31] also uses the CSI data to perform an analysis on popular neural networks like (DNN, CNN, and KNN). The work in [41] proposes an early-exit deep neural network strategy for fast indoor localization on mobile devices. These prior works motivate the use of ML-based techniques for indoor localization. LearnLoc [42]

extends the K-Nearest-Neighbor (KNN) algorithm by matching fingerprints with a lightweight Euclidean distance-based metric. Surprisingly, the authors of [43] conducted an error analysis to better understand the advantages of distance calculations. The primary goal was to compare Euclidean versus Manhattan distance-based metrics for matching fingerprints. The research shows that using a Euclidean distance-based metric to match fingerprints is more accurate. However, none of these efforts address real-world challenges related to device heterogeneity.

The work in [26] performed a comparative study on the behavior of classification and regression-based ML models and their ability to deal with device heterogeneity. In [44], an unsupervised learning mechanism with Generative Adversarial Networks (GAN) was used for data augmentation with deep neural networks. However, this work does not address localization with noisy fingerprint data (e.g., fluctuating RSS data from different smartphones). The work in [45] employed a deep autoencoder for dealing with noisy fingerprint data and device heterogeneity. The deep autoencoder was used along with traditional K-Nearest-Neighbor (KNN) classification models, but results for the approach across smartphones are not promising. Calibration-free approaches were proposed that used Signal Strength Difference (SSD) and Hyperbolic Location Fingerprints (HLF) with pairwise ratios to address heterogeneity [46]. However, both these approaches also suffer from slow convergence and high errors across larger sets of diverse smartphones.

The recent works ANVIL [28], SHERPA [17], CNNLoc [47], and WiDeep [48] improve upon the efforts mentioned above, to more effectively cope with device heterogeneity while maintaining lower localization errors. In, [17] a multi-headed attention neural network-based framework was proposed for indoor localization. The work in [47] combined KNN and deep neural networks for fingerprint classification. In [48], CNNs were used for regression-based localization prediction.

The work in [30] employed a stacked autoencoder for data augmentation and a Gaussian process classifier for localization classification. The ML models and augmentation techniques in all of these works aim to improve resilience to device heterogeneity while minimizing localization error.

As [17], [30] represent the state-of-the-art heterogeneity-resilient indoor localization efforts, we modeled these frameworks and used them to contrast against our framework. [29] performs an external analysis on developing long-term localization models using Siamese neural networks. The work in [49] focuses on incorporating secure indoor localization techniques using CNNs for AP security. These extended works [47], [48] may serve as the basis for future improvements in the VITAL framework. Our proposed VITAL framework integrates a novel data augmentation approach with an enhanced vision transformer neural network to achieve promising results for resilience towards device heterogeneity, while maintaining extremely low localization errors, as shown in Section 3.6.

3.4. VISION TRANSFORMERS OVERVIEW

Transformers, first introduced in the well-known paper "Attention is All You Need" [22] in 2017, have spread widely in the field of Natural Language Processing, quickly becoming one of the field's most widely used and promising architectures. With the paper "An image is worth 16x16 words," [22] Vision Transformers were then adapted for tasks in Computer Vision in 2020. The idea is to break down input images into a series of patches that, once transformed into vectors, are seen as words in a standard transformer. A transformer is a powerful deep neural network model that relies on the self-attention mechanism [46]. The self-attention mechanism mimics human cognitive attention, enhancing some parts of the input data while diminishing other parts, via a differential weighting approach. The motivation for doing so is to force the neural network to

devote more focus to the small, but important, parts of the data. Learning which part of the data is more important than another depends on the context, which is trained by gradient descent. Transformers are today being widely used in Natural Language Processing (NLP) tasks.

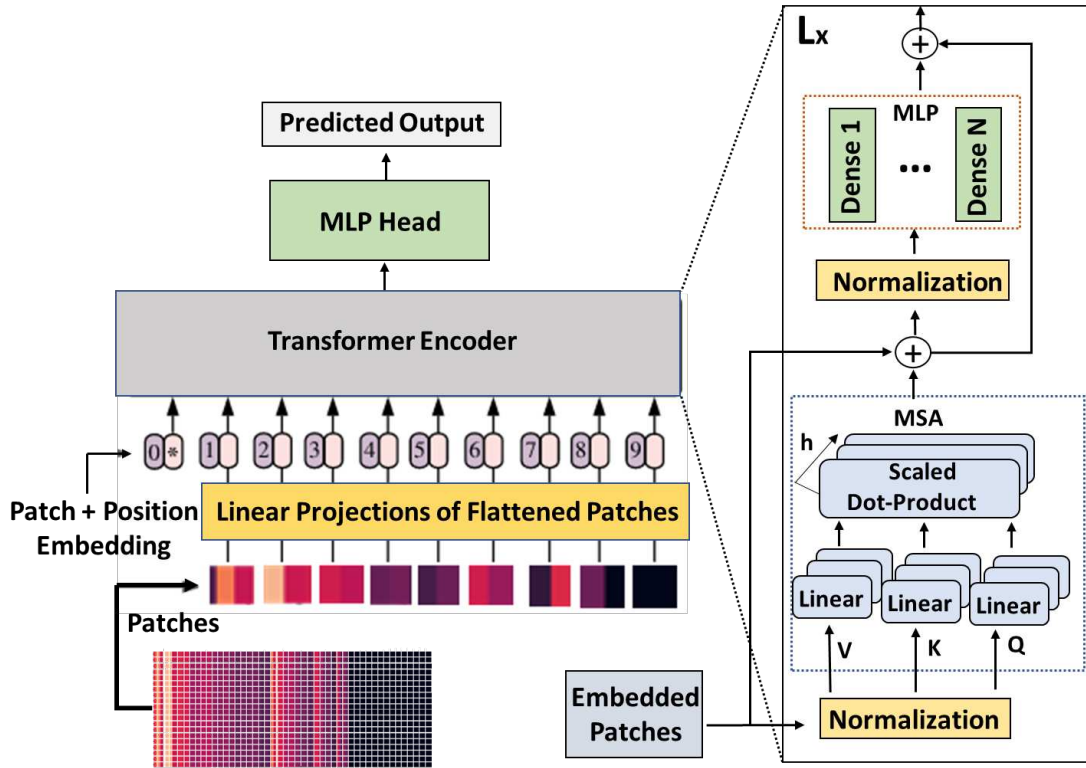


Figure 15: Overview of Vision Transformer Neural Network

Since the original transformer was proposed, many efforts have attempted to go beyond NLP and apply it to computer vision tasks. Unfortunately, the naive application of self-attention to images requires that each pixel attends to every other pixel. With quadratic cost in the number of pixels, this does not scale to realistic input sizes. The vision transformer (ViT) model [22] was recently proposed as a scalable approach to applying transformers to image processing. ViT improves upon the traditional transformer model which lacks some of the image-specific inductive biases (such as translation equivariance and locality) that make convolutional neural networks

(CNNs) very effective for computer vision tasks. The principal paper [22] effectively prepares a Transformer encoder on ImageNet, accomplishing generally excellent outcomes in contrast with convolutional networks. The vision transformer is essentially an improvement over traditional transformer. Traditional transformers miss the mark on inductive predispositions of CNNs, like interpretation invariance and a locally restricted receptive field. Interpretation invariance implies that you can perceive an element (for example object) in a picture, in any event, when its appearance or position changes. Interpretation in computer vision suggests that each picture pixel has been moved by a proper sum in a specific direction. Convolution being a linear local operator whose values depend on the neighboring pixel data indicated by the kernel possesses a limitation when an object in the image changes position. The traditional transformer also faces the same limitation as that of Convolution neural networks. The architecture of the vision transformer overcomes this issue by being permutation invariant.

An overview of the ViT model is depicted in Figure 15. ViT consists of three main components: 1) position embedding of patches, 2) transformer encoder block, and 3) a Multi-Layer Perceptron (MLP) head. The first component of ViT creates patches from an image, essentially dividing the input image into small chunks and then positionally encoding each chunk to retain the order of the patches created. These positionally encoded patches are referred to as embedded patches and allow the ViT to be permutationally invariant. The embedded patches are subsequently sent as inputs to the transformer encoder, which is the second component. The layers of the transformer encoder are shown in Figure 15 and consist of L alternating layers of a multi-headed self-attention (MSA) and an MLP block, in addition to layer normalizations, and residual connections. In the MSA block h denotes the number of heads. The output of the transformer is sent to the third component, which is an MLP head that contains two layers with a GELU non-linearity for classification.

3.5. VITAL FRAMEWORK

The Figure 16 depicts an overview of our proposed VITAL framework, which adapts (and improves) vision transformers for the problem of indoor localization. The proposed framework is divided into two phases: offline (shown with red arrows) and online (shown with blue arrows). During the offline phase, RSS fingerprints are collected at each RP using different smartphones across multiple buildings. At each RP, we use three fingerprint values to generate a 1D image with three channels using a custom RSS image creator. This module converts each AP's three RSS readings into a pixel. Thus, a pixel represents an AP's three RSS values, and the fingerprint vector at an RP is a 1D image composed of RSS readings from all APs. A data augmentation module (DAM) is used to pre-process the 1D image, which is then represented as 2D images. These images are then fed into a vision transformer model, which learns the relationship between fingerprints and RP locations. The user requesting their location captures RSS fingerprints from their smartphone at an unknown location during the online phase. The RSS fingerprints are pre-processed with DAM and represented as an image, as done in the offline phase. The image is then sent to the trained vision transformer to predict the location.

In the following subsections, we describe the two main components of our framework: DAM and the vision transformer.

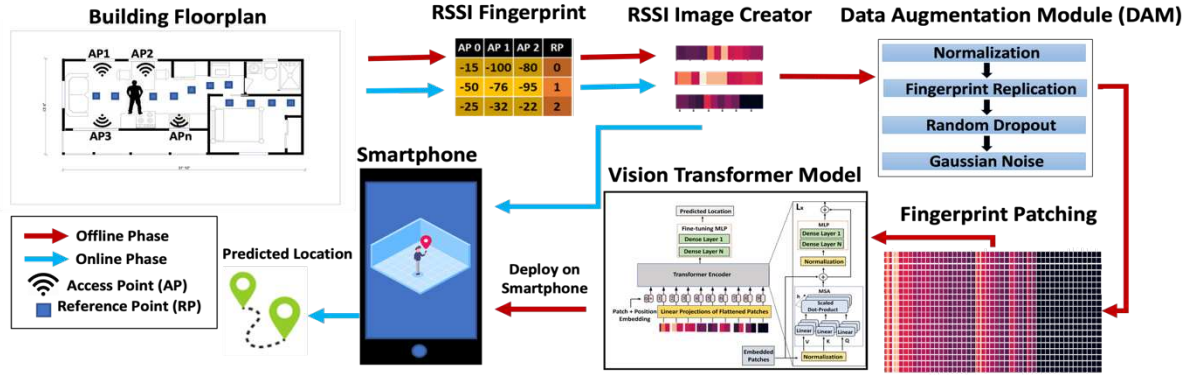


Figure 16: An overview of the proposed device heterogeneity-resilient VITAL indoor localization framework.

3.5.1. DATA AUGMENTATION MODULE (DAM)

Our data augmentation module (DAM) is responsible for preparing the input data to help improve the sensitivity of location prediction and make it calibration-free. DAM consists of four stages as shown in Figure 16: data normalization, fingerprint replication, random dropout, and gaussian noise.

The first stage involves normalizing the input data. This is a critical step because it ensures that each pixel has a similar distribution, allowing for faster convergence during training. The layer standardizes the fingerprint's features during training to achieve smoother gradients and better generalization. The second stage replicates the fingerprint data to combine the augmented and original features into a single image. The replication is carried out in such a way that the input 1D image of size $1R$ (where R represents the number of RPs) is augmented and represented as a 2D image of size RR . We investigated various image augmentation sizes to determine the best value, and the results are presented in Section 3.6.2. The third and fourth steps involve the addition of random dropout and Gaussian noise. The random dropout is used to randomly drop some APs or features in the input data to represent missing APs and allow the model to be robust to missing APs when making location predictions. The Gaussian noise is used to infill the dropped features

with some random noise to represent different AP visibilities. This allows capturing the effect of fluctuating RSS values due to dynamic environmental factors in indoor locations and makes the trained model more robust to RSS variations.

In our framework, DAM is applied after the fingerprint capture step. This module can be integrated into any ML framework to improve robustness of indoor localization. In fact, we demonstrate in Section 3.6.4 that DAM can have a notable positive impact for many indoor localization frameworks from prior work.

3.5.2. VISION TRANSFORMER

The output from DAM is sent to a vision transformer model that we modify and adapt for the indoor localization problem domain. As will be explained below, we changed several of the layers from the vision transformer model proposed in [48], as well as the image formation procedure, MSA input parameters, and MLP layers in the transformer encoder and classification head.

Post augmentation from DAM, we now have a 2D image of size $R \times R$. This image is sliced into small patches of size $P \times P$. We explored the impact of different patch sizes, with results presented in Section 3.6.2. The number of patches per image is represented as N , where N is calculated by dividing the area of the input image ($H \times W$) by the area of the patch sizes, i.e., $N = (H \times W) / (P \times P)$. The N patches serve as an input sequence to the vision transformer. As the model expects a constant vector of size N for all its layers, to satisfy this requirement, we flatten the patches and map them as a linear trainable projection. These projections are then sent to the transformer encoder block.

The self-attention sub-block in the transformer encoder calculates a weighted average for each feature in the projections from the input image. The weights are proportional to the similarity score

between each feature. The attention mechanism in the sub-block is modeled such that it extracts values (V) with associated keys (K) for a given query (Q). Through this, we calculate attention as:

$$Attention(Q, K, V) = softmax(Dot Score)V \quad (4)$$

$$Dot Score = \left(Q K^T / \sqrt{d_k} \right) \quad (5)$$

$$Q = XW^Q, K = XW^K, V = XW^V \quad (6)$$

where d_k is the dimensionality of the key, X is the input, and W^Q , W^K , and W^V are the trainable weight matrices for Q , K , and V , respectively. In our framework, Q is the patched images, K is the one-hot encoded positions of each patch, and V is the one-hot encoded RP locations of the input image. The higher the dot product score, the higher the attention weights will be, which is why it is considered a similarity measure. Using this information, we implement a multi-headed self-attention (MSA) mechanism to extract information from different input representations. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this. The multi-headed attention function is expressed as:

$$MultiHead(Q, K, V) = Concat(h_1, h_2, \dots, h_n)W^O$$

$$where h_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (7)$$

We determine the optimal number of MSA heads for our problem via sensitivity analysis, described in Section 3.6.2. In addition to the MSA sub-block, the encoder also contains an MLP sub-block which consists of two dense neural network layers with a GELU non-linearity. The output from the MSA sub-block is concatenated with outputs from the MLP sub-block to restore any lost features. The encoder also uses layer normalization before each MSA and MLP sub-block to normalize each of the inputs independently across all features. The encoder outputs a vector of values that corresponds to the location class of the input RSS image.

The final step is to fine-tune the output from the transformer encoder. This is done to make the indoor localization framework calibration-free. To fine-tune the output vectors from the transformer encoder, a fine-tuning MLP block is used. This block consists of a series of MLP layers. We explored the optimal number of dense layers within the fine-tuning MLP block for our problem via sensitivity analysis, described in Section 3.6.2. The last dense layer contains as many neurons as the number of RPs.

When doing inference with RSS values gathered by the user at an unknown location, the trained model was first deployed on smartphones. The RSS fingerprints gathered by the smartphone's Wi-Fi transceiver are accessible, pre-processed, and then displayed as images, just like they were done during the offline phase. Our vision transformer neural network model receives the requested image and uses it to forecast the unknown location.

The VITAL framework is set to be calibration-free by group training the neural network model. The group training combines RSS fingerprint data from different smartphones for RPs. This approach allows the model to learn the vagaries of RSS visibility across different smartphones. In the next section, we present our experimental results, where we also discuss how the trained model

generalizes and performs on new smartphones that were not part of the pool of smartphone devices used to collect the RSS training data to train our vision transformer model.

3.6. EXPERIMENTS

3.6.1. EXPERIMENTAL SETUP

The VITAL framework was evaluated in four different buildings. The RSS fingerprints are captured along paths in each of the buildings, as shown in Figure 17. The samples were collected using a granularity of 1 meter between each RP. The blue dots in Figure 17 represent the RPs for fingerprint collection. The path lengths varied from 62 meters to 88 meters. Each building also had a different number of Wi-Fi access points (APs). Note that only a subset of the APs present in a building were visible at each of the RPs in that building. Each of the buildings also had a very different material composition (wood, metal, concrete) and was populated with diverse equipment, furniture, and layouts, creating different environments in which to evaluate our framework as well as other heterogeneity-resilient localization frameworks from prior work.

For our experiments, we made use of six distinct smartphones from different manufacturers, to capture RSS data across the RPs in the four buildings and train the indoor localization frameworks. Table 2 summarizes the details of these smartphones. The devices operated under the latest OS available at the time of this experiment. We refer to this set of devices as ‘Base’ devices. The RSS data captured across the building consisted of five RSS data samples captured at an RP for each of the six smartphones, across the four buildings. These five values were reduced to three by capturing their min, max, and mean values, and used to create the three channels for each element (pixel) in the fingerprint vector that is input to our VITAL framework. The RSS data captured was

split into two datasets: training (approximately 80%) and testing (approximately 20%), with results reported for the testing dataset.

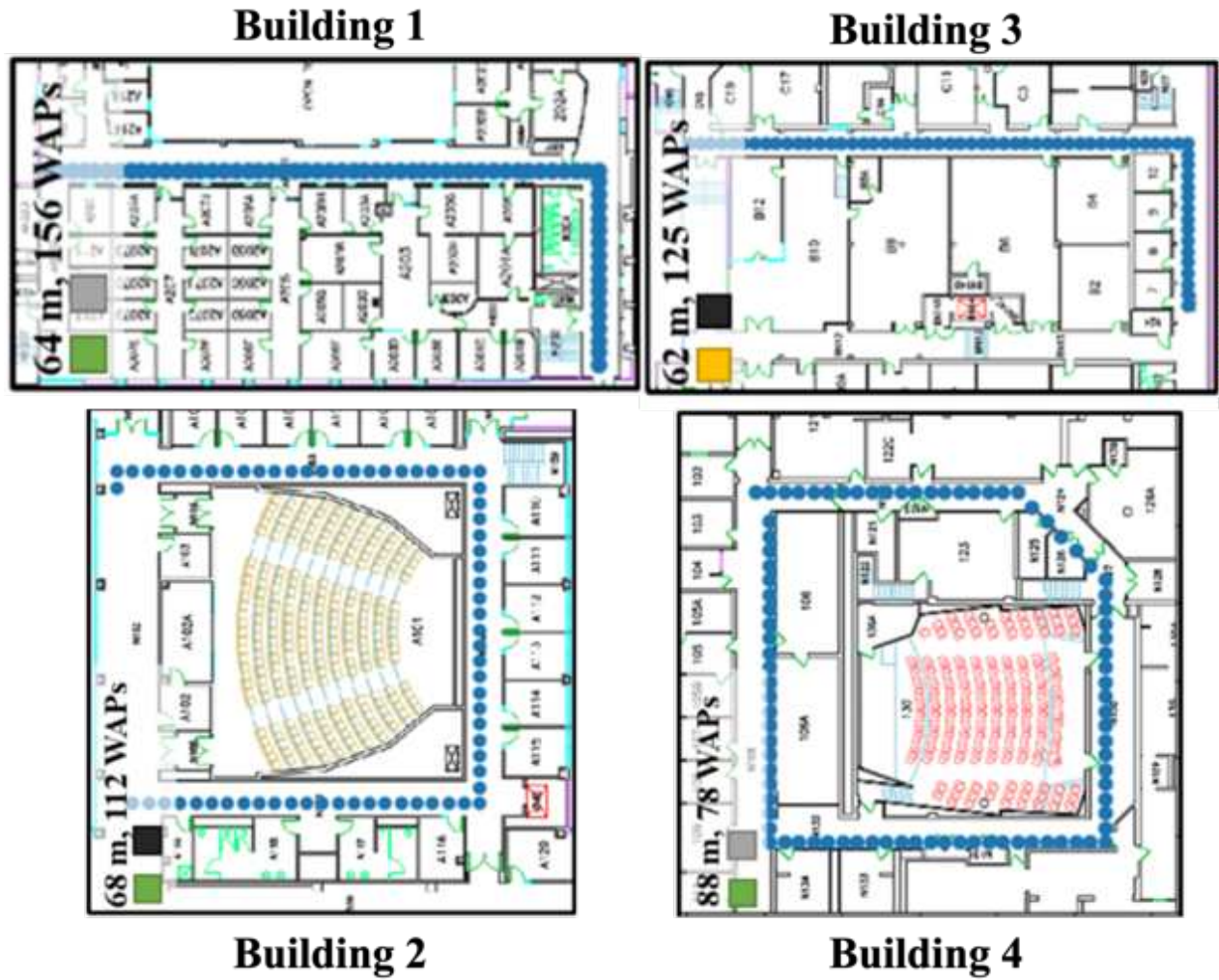


Figure 17: Benchmark indoor paths that were used for fingerprint collection. We collected RSS data across four buildings, with varying path lengths. The blue dots indicate reference points (RP) at which the RSS reading were taken. Six different smartphones were used to capture RSS data at each RP.

Table 2: Smartphones used for evaluation (Base devices).

Manufacturer	Model	Acronym	Release Date
BLU	Vivo 8	BLU	2017
HTC	U11	HTC	2017
Samsung	Galaxy S7	S7	2016
LG	V20	LG	2016
Motorola	Z2	MOTO	2017
Oneplus	OnePlus 3	OP3	2016

3.6.2. HYPERPARAMETER EXPLORATION

The VITAL framework involves multiple hyperparameters. While an exhaustive search across all possible values for all hyperparameters was not practically possible, we focused on analyzing four critical hyperparameters across the pre-processing and transformer architecture design, as part of a sensitivity analysis.

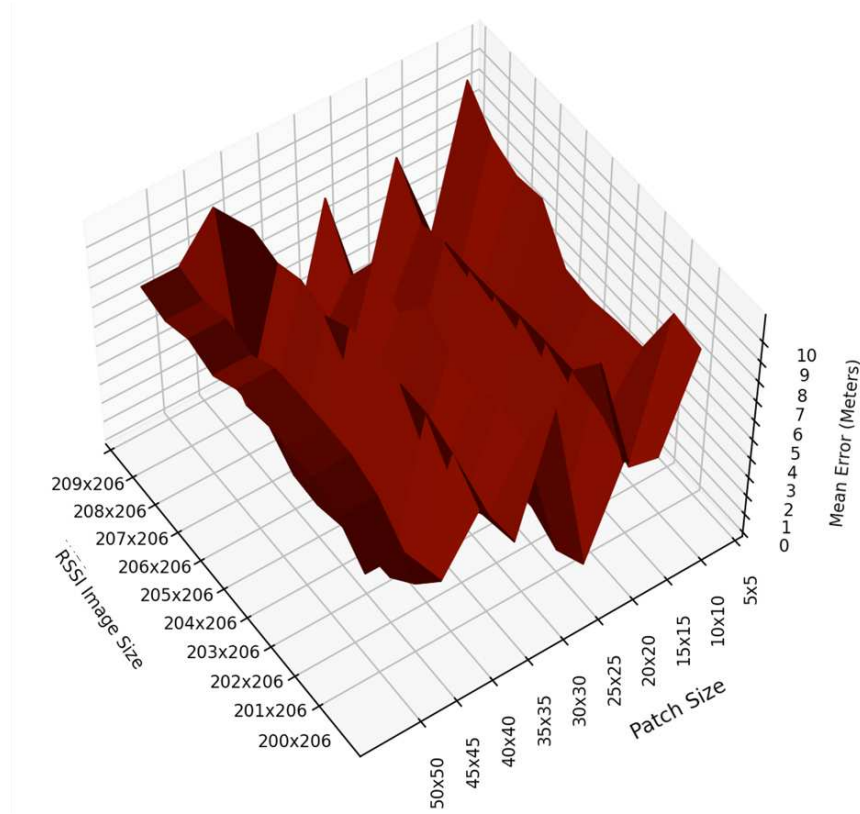


Figure 18: Surface plot showing the impact of patch size and RSS image sizes on the mean indoor localization error.

In the first analysis, we focused on the input pre-processing phase in VITAL, within DAM. Figure 18 depicts the impact of different RSS fingerprint image sizes and patch sizes on the mean indoor localization error with the VITAL framework. From the results, we conclude that an RSS fingerprint image size of $R \times R = 206 \times 206$ and patch size of $P \times P = 20 \times 20$ results in the least mean localization error. Smaller patch sizes under-perform as they lead to a greater number of patches being created, which tends to over-fit the data, whereas larger patch sizes tend to under-fit the data. The RSS fingerprint image sizes had relatively less impact on the localization error. However, we did note that image sizes that led to the creation of partial patches at the boundaries, led to the discarding of some features, which tended to reduce accuracy.

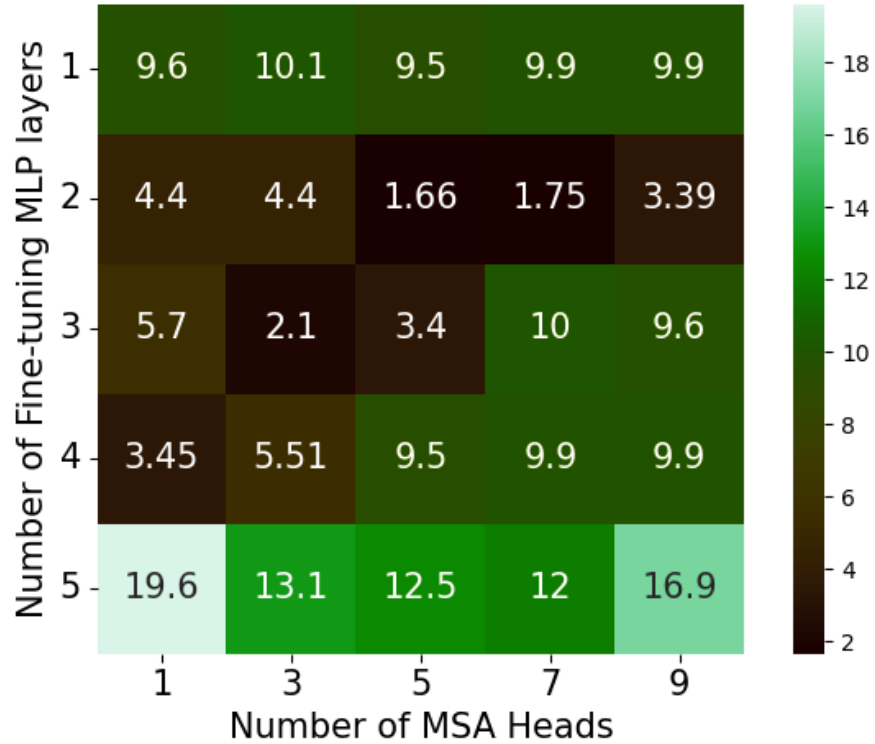


Figure 19: Impact of number of fine tuning MLP layers and number of MSA heads in our vision transformer model on mean indoor localization error.

We further explored two hyperparameters of the vision transformer encoder block used within the VITAL framework. We analyzed the impact of the number of multi-headed self-attention (MSA) heads and dense layers in the fine-tuning MLP block on the mean indoor localization error. Figure 19 shows a heatmap of this exploration. We observe that an MSA head count of five and two fine-tuning MLP layers show the least mean indoor localization error (shown in meters in Figure 19). A lower number of MLP layers tends to under-fit the data whereas a higher number of MLP layers tends to over-fit the data. We also observe that the number of MSA heads has a significant impact on the model’s performance. A higher head count tends to over-fit the data.

Based on the results from our hyperparameter analysis, we finalized the configuration of the VITAL framework that was used to compare with frameworks from prior work (next Section). We used an RSS fingerprint image size of $R \times R = 206 \times 206$, patch size of $P \times P = 20 \times 20$, number of transformer encoder blocks $L=1$, number of MSA heads = 5, number of dense layers in the MLP sub-block within the encoder = 2 (with 128 and 64 units, respectively), and the number of dense layers in the fine-tuning MLP block = 2 (with 128 and *num_classes* units, respectively). The total number of trainable parameters in the transformer model was 234,706. This compact model was able to perform inference within ~ 50 ms, making it amenable to deployment on memory-constrained and computationally limited embedded and IoT platforms.

3.6.3. COMPARISION WITH STATE-OF-THE-ART

We compared our VITAL framework with four state-of-the-art frameworks for heterogeneity-resilient indoor localization: ANVIL [17], SHERPA [47], CNNLoc [48], and WiDeep [30]. These frameworks have shown the most promising results for device heterogeneity tolerance during indoor localization. ANVIL [17] employs a multi-headed attention mechanism and a Euclidean

distance-based matching approach to achieve device heterogeneity tolerance. In our proposed VITAL framework, we enhance the attention mechanism used in ANVIL using multi-headed self-attention as part of the vision transformer encoder, as explained earlier. CNNLoc [48] employs a combination of stacked autoencoder (SAE) and 1D CNN to tackle device heterogeneity. The SAE is used as an augmentation to better deal with noisy fingerprint data. In VITAL, we make use of the more powerful DAM, described earlier, to deal with noisy data. Lastly, SHERPA [47] employs a KNN algorithm enhanced with DNNs and WiDeep [30] utilizes a Gaussian Process Classifier enhanced with an SAE, respectively.

Figure 20 shows a color-coded comparison of VITAL with these state-of-the-art frameworks. We can observe that WiDeep consistently shows high mean errors across most buildings, making it the worst performing framework. One possible explanation for this is the effect of the de-noising SAE, which very aggressively creates noisy reconstructions of the inputs in a manner that makes it difficult for the classifier to arrive at accurate predictions. CNNLoc also employs SAE but shows better results compared to WiDeep, as its algorithms can deal with noisy data more effectively. Interestingly, CNNLoc fails to localize accurately in less noisy environments, such as that found in Building 4. ANVIL and SHERPA show better results than WiDeep and CNNLoc, with a few outliers. Our proposed VITAL framework shows overall much better results than these four state-of-the-art frameworks as it uses a more powerful learning engine (transformer) and is better able to exploit heterogeneous fingerprints captured across smartphones than the other frameworks. To better visualize the performance of these frameworks, Figure 20 shows a box plot with minimum, maximum, and mean errors across all buildings. Our VITAL framework has the least minimum, maximum, and mean errors compared to all other frameworks. VITAL has the least mean error (1.18 meters), followed by ANVIL (1.9 meters), SHERPA (2.0 meters), CNNLoc (2.98 meters),

and WiDeep (3.73 meters). Thus, VITAL achieves improvements ranging from 41% to 68%. VITAL also has the lowest maximum error (3.0 meters), followed by ANVIL (3.56 meters), SHERPA (6.22 meters), CNNLoc (4.58 meters), and WiDeep (8.2 meters).

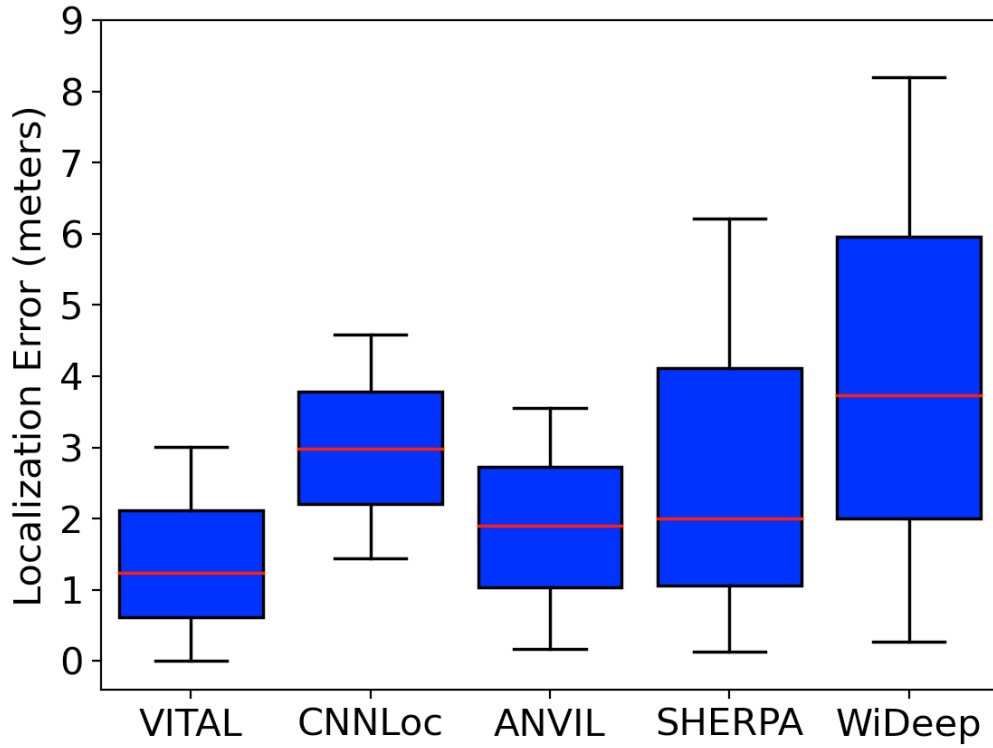


Figure 20: Min (lower whisker), Mean (red bar), and Max (upper whisker) error across all buildings for comparison frameworks w/ base devices

3.6.4. EFFECTIVENESS OF DAM

The Data Augmentation Module (DAM) is a key component of the proposed VITAL framework, with broad applicability to a variety of indoor localization frameworks. We incorporated DAM into the four comparison frameworks to assess its impact. Figure 21 depicts a slope graph of mean error for all frameworks with and without DAM. VITAL can achieve lower mean localization errors with DAM integration than without it. Furthermore, when combined with

ANVIL, SHERPA, and CNNLoc, three of the four cutting-edge frameworks we compare, DAM shows significant improvements. WiDeep exhibits higher mean errors when DAM is included, indicating that it is prone to overfitting. Thus, DAM can significantly improve localization accuracy predictions for some frameworks, such as ANVIL, SHERPA, and CNNLoc, as well as our VITAL framework proposed in this chapter.

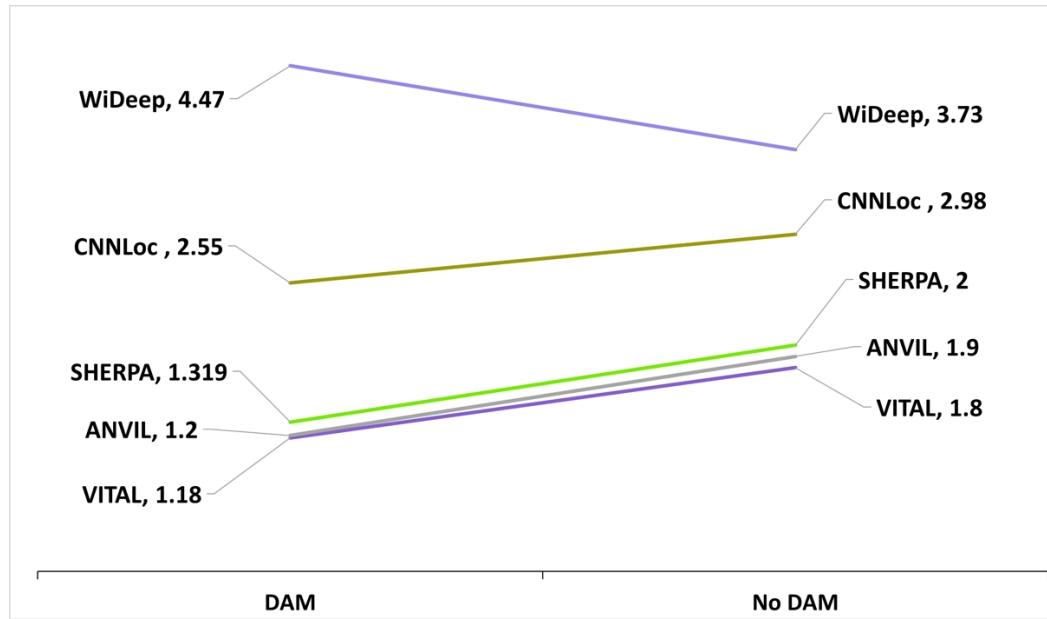


Figure 21: Slope graph of the impact of DAM on all frameworks.

3.6.5. EXPERIMENTS ON NEW SMARTPHONES

To further evaluate the generalizability of our VITAL framework, we conducted another experiment by deploying the frameworks on three new smartphone devices on which the frameworks were not trained. The new devices, referred to as ‘extended devices’, are shown in Table 3.

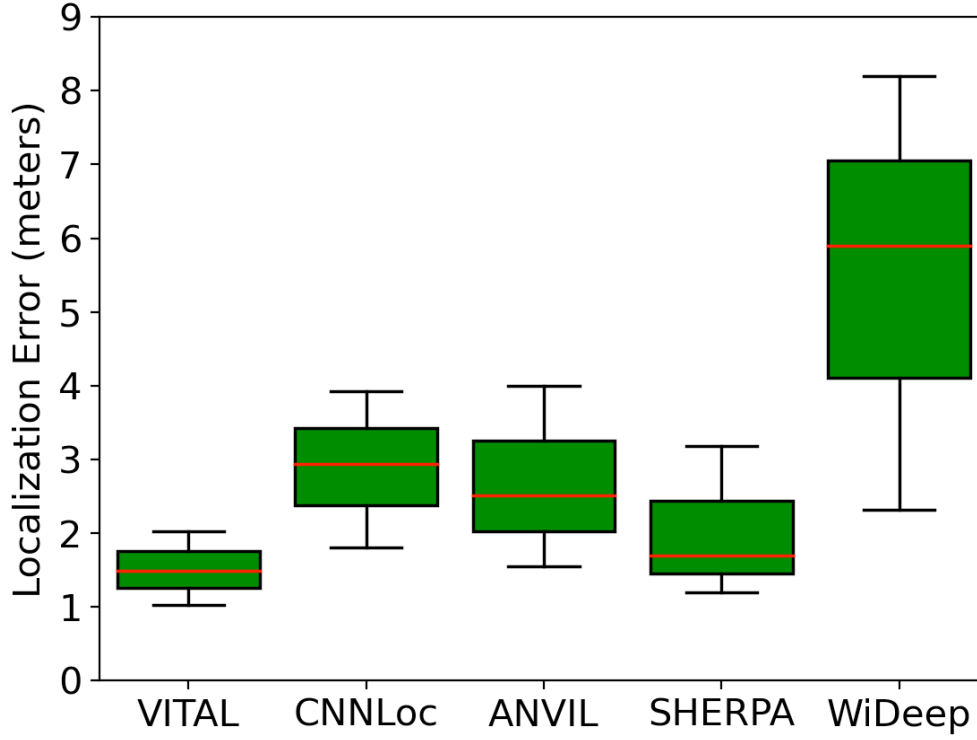


Figure 22: Min (lower whisker), Mean (red bar), and Max (upper whisker) error across all buildings for comparison frameworks w/ extended devices.

Table 3: Smartphones used for evaluation (Extended devices).

Manufacturer	Model	Acronym	Release Date
Nokia	Nokia 7.1	NOKIA	2018
Google	Pixel 4a	PIXEL	2020
Apple	iPhone 12	IPHONE	2021

Figure 22 shows a box plot with minimum, maximum, and mean errors across all buildings for the comparison frameworks, across the three extended devices. From the Figure, we can conclude that VITAL outperforms all frameworks significantly. VITAL has the least mean error (1.38 meters), followed by SHERPA (1.7 meters), ANVIL (2.51 meters), CNNLoc (2.94 meters), and WiDeep (5.90 meters). Thus, VITAL achieves improvements ranging from 19% to 77%. VITAL also has the least maximum error (3.03 meters), followed by SHERPA (3.18 meters), ANVIL (4.0

meters), CNNLoc (3.92 meters), and WiDeep (8.20 meters). Thus, VITAL enables superior device heterogeneity tolerance and localization accuracy, even on new and unseen devices that it has not been trained on.

3.7. CONCLUSION

In this chapter, we introduced VITAL, a novel framework that is resistant to device heterogeneity during indoor localization. VITAL was tested against four cutting-edge frameworks for device heterogeneity-resilient indoor localization in four different buildings and with nine different smartphone devices. When compared to the best-performing frameworks from previous work, VITAL improves mean localization error by 41% to 68%. VITAL also exhibits excellent generalizability across new smartphone devices, with mean localization error improvements ranging from 19% to 77% when compared to the best-performing frameworks from previous work. VITAL's calibration-free approach makes it a popular framework for indoor localization in real-world settings.

CHAPTER 4

STELLAR: SIAMESE MULTI-HEADED ATTENTION NEURAL NETWORKS FOR OVERCOMING TEMPORAL VARIATIONS AND DEVICE HETEROGENEITY WITH INDOOR LOCALIZATION

Smartphone-based indoor localization has emerged as a cost-effective and accurate solution to localize mobile and IoT devices indoors. However, the challenges of device heterogeneity and temporal variations have hindered its widespread adoption and accuracy. Towards jointly addressing these challenges comprehensively, we propose STELLAR, a novel framework implementing a contrastive learning approach that leverages a Siamese multi-headed attention neural network. STELLAR is the first solution that simultaneously tackles device heterogeneity and temporal variations in indoor localization, without the need for re-training the model (re-calibration-free). Our evaluations across diverse indoor environments show 8-75% improvements in accuracy compared to state-of-the-art techniques, to effectively address the device heterogeneity challenge. Moreover, STELLAR outperforms existing methods by 18-165% over 2 years of temporal variations, showcasing its robustness and adaptability.

4.1. INTRODUCTION

In recent years, indoor localization has garnered significant attention due to its wide range of applications, including healthcare, asset tracking, navigation systems, location-based advertising, and much more [60]. The ability to accurately determine the position of mobile and IoT devices within indoor environments has the potential to revolutionize various industries and enhance user

experience [61]. However, achieving high-precision indoor localization remains a challenging task, due to the inherent complexities and dynamic nature of indoor environments.

To achieve high-precision indoor localization, many researchers in the domain have turned to Wi-Fi received signal strength (RSS) based indoor localization [62]. This method leverages using Wi-Fi RSS from Wi-Fi routers or access points (APs) to determine the position of mobile and IoT devices within indoor environments. Fingerprinting model-based systems [3] have gained significant popularity due to their robustness and compatibility with existing Wi-Fi infrastructure in complex indoor environments [37]. Despite their effectiveness, these systems have limitations. Inaccuracies may arise due to signal variability caused by obstacles, signal interference, and other factors, such as signal attenuation, multi-path fading, and shadowing, which can distort signals and result in inaccurate location estimates [63]. These limitations underscore the importance of developing robust localization methods to overcome these challenges.

Fingerprinting model-based systems typically involve a database of Wi-Fi signal patterns, commonly known as “fingerprints”, are captured across the indoor space. This database allows for more robust and accurate positioning or localization [64], making it an increasingly preferred choice in recent indoor localization developments and competitions (e.g., [35], [75]), avoiding the need for additional hardware deployment [65]. However, changes in the indoor environment such as furniture and equipment rearrangement or the addition of new obstacles can render the stored fingerprints less accurate, necessitating frequent updates. Moreover, fingerprinting-based systems may face challenges in handling large-scale deployments, as the collection and management of extensive fingerprint databases can become cumbersome [64], [65]. Hence, there is a need for a robust approach for fingerprinting-based localization.

Machine learning (ML)-based fingerprinting approaches can be used to learn patterns and relationships from the collected RSS fingerprint data, enabling accurate positioning of devices in indoor environments [66]. ML-based fingerprinting systems consists of two phases: The offline (training) phase and the online (inferencing) phase [67]. During the offline or training phase, a dataset of RSS fingerprints is collected from known locations within the indoor environment. This dataset serves as the training data for the ML model. Once the ML model is trained, it can be used in the online or inference phase for real-time positioning of devices [37], [47].

Unfortunately, ML-based fingerprinting systems face many challenges arising from the phenomenon of device heterogeneity. Mobile devices from different manufacturers use varying Wi-Fi chipset configurations, leading to different captured RSS patterns across devices that are at the same location. This discrepancy in captured RSS makes it difficult for the ML model to accurately predict the device's location [118]. Furthermore, indoor environments are not only subject to device heterogeneity but are also influenced by various sources of external noise [45], [73]. Additionally, the often-overlooked temporal variations caused by changes in APs and indoor environments over months to years exacerbates the complexity of the localization process [45], [73]. APs may be added, dropped, or turned off due to maintenance, power outages, or other reasons, leading to temporal variations that significantly impact the availability and quality of RSS signals [74]. Existing methods often struggle to adapt to these changes and maintain accuracy in the face of such challenges.

4.2. CONTRIBUTIONS

In this chapter, we introduce STELLAR, an innovative framework that pioneers addressing both device heterogeneity and temporal resilience in indoor localization, all without requiring frequent

model re-training (“re-calibration-free”). Unlike many ML-based fingerprinting methods, which typically require re-training to handle dynamic indoor challenges. STELLAR excels by not needing frequent recalibration or updates, ensuring consistent performance in the face of evolving indoor environments. By leveraging contrastive learning, an ML technique that maximizes similarity between similar RSS samples and minimizes it between dissimilar ones, STELLAR is able to address complex indoor localization challenges, resulting in a robust and long-term indoor localization solution. The novel contributions of this work are:

- We propose a framework called STELLAR to address both device heterogeneity and temporal variations, for accurate and long-term indoor localization.
- We develop a contrastive learning and re-calibration-free localization approach based on a Siamese multi-headed attention neural network ML model.
- Towards the goal of improving both device heterogeneity resilience and temporal invariance, we identify and adapt data modulation methodologies for training the proposed ML model.
- By collecting fingerprint(s) with multiple heterogeneous devices across buildings and time (few seconds to years), we create benchmarks to test the localization accuracy of STELLAR against the state-of-the-art.
- We prototype our re-calibration-free indoor localization framework, deploy it on diverse smartphones, and evaluate it under real-world settings.

In the following sections, we provide a detailed overview of the related work in indoor localization, discuss the methodology and architecture of the proposed STELLAR framework, present our experimental setup and results, and conclude with the implications of this research.

4.3. RELATED WORK

Indoor localization has been studied for several decades now, with Wi-Fi fingerprinting based indoor localization gaining prominence in recent competitions by Microsoft [35] and NIST [75]. Wi-Fi fingerprinting methods have shown better accuracy compared with other approaches, making fingerprinting a particularly promising approach for indoor positioning [65]. Simultaneously, advancements in the computational capabilities of smartphones have played a pivotal role in empowering them to execute deeper and more powerful ML models, which have the potential to enhance indoor localization performance. As a result, researchers have been actively exploring the integration of ML with Wi-Fi fingerprinting to improve indoor localization accuracy and efficiency in recent years.

Various ML algorithms, including support vector machines (SVM) [69], K-nearest neighbors (KNN) [76], random forest (RF) [77], and neural networks such as deep neural networks (DNN) [78] and convolutional neural networks (CNN) [82], have been employed to capture intricate non-linear relationships within indoor localization data. These ML algorithms, with their ability to handle complex patterns and relationships, have proven to deliver higher accuracies compared to traditional non-ML based fingerprinting-based solutions. However, despite the integration of ML techniques, fingerprinting-based methods may encounter challenges in handling device heterogeneity (fluctuations in RSS fingerprints due to various devices) and temporal variations (changes in APs over months to years indoors), as discussed earlier.

To address the device heterogeneity challenge, researchers have made use of advanced neural network-based ML algorithms in recent work. For instance, ANVIL [45] utilized a multi-headed attention neural network combined with domain-specific augmentation to address device heterogeneity in challenging indoor environments with signal reflections and obstructions. The use

of attention mechanisms enhances the localization process in ANVIL by selectively focusing on relevant signal patterns. VITAL [80] introduces a method for converting Wi-Fi fingerprints into equivalent images, enabling the use of the vision transformer model with a unique image-based data augmentation module to tackle the challenge of device heterogeneity. The inclusion of an attention neural network within the vision transformer further improves localization accuracy by focusing on relevant signal patterns. VITAL demonstrates comparable performance to ANVIL [45] when addressing device heterogeneity for indoor localization. WiDeep [17] incorporates an autoencoder-supported Gaussian process classifier to tackle the challenges of device heterogeneity. By utilizing an autoencoder, WiDeep learns condensed and informative representations of Wi-Fi signal patterns, mitigating the impact of device variations and enhancing the model's generalization capabilities. Subsequently, the Gaussian process classifier harnesses these learned representations to effectively model the intricate non-linear relationships in the data. SANGRIA [82] employs a stacked autoencoder to address the challenges of device heterogeneity. SANGRIA outperforms WiDeep [17] due to its stacked autoencoder being able to learn hierarchical representations of Wi-Fi signal patterns more efficiently, enabling improved handling of device variations. While all of these works show promising potential, they do not address the challenges associated with temporal variations (changes in APs and indoor environments over months to years indoors), resulting in limited resilience over time.

A few recent works have begun to address temporal variations during indoor localization. LT-KNN (long-term KNN) [83] addresses situations where multiple APs are removed during the online phase (over a 15-month span of time), by incorporating regular re-training, thereby improving the performance of KNN in such scenarios. However, the need for frequent re-training due to sensitivity to temporal variations is a significant overhead. Another approach in [84] utilizes

a simple Siamese neural network using fully connected layers to address the challenges associated with temporal variations. However, like LT-KNN [83], it suffers from the drawback of requiring frequent re-training. This re-training (or re-calibration) phase is necessary to adapt ML models to the dynamic changes in the indoor environment. But re-training requires collecting new data, which can be time-consuming, leading to temporary service disruptions and potential model overfitting or underfitting. These factors hinder the system's real-time responsiveness and adaptability, making it less practical for long-term indoor localization deployments.

Our proposed STELLAR framework described in this chapter is a re-calibration-free framework that does not require re-training over time, while being able to simultaneously address both device heterogeneity and temporal variations. This framework builds upon our prior work ANVIL [45] that was presented at the 2022 IEEE IPIN conference. STELLAR improves upon ANVIL in several ways: (i) STELLAR addresses both device heterogeneity and temporal variations, whereas ANVIL only focuses on device heterogeneity, (ii) STELLAR utilizes a new gradient boosting approach to enhance resilience towards heterogeneity, leading to improved localization performance compared to ANVIL, (iii) unlike ANVIL, STELLAR is a re-calibration-free model, by leveraging a more sophisticated Siamese multi-headed attention approach for enhanced localization performance, and (iv) we add multiple new experiments and incorporate newer smartphone devices during the online phase to more comprehensively evaluate STELLAR's effectiveness for robust indoor localization.

4.4. ANALYSIS OF DEVICE HETEROGENEITY VARIATIONS

Wi-Fi fingerprinting based indoor localization systems heavily rely on Wi-Fi RSS values captured by mobile devices for their accuracy and reliability. However, the presence of device

heterogeneity introduces significant variations in these RSS readings, which can negatively impact localization results. To understand the implications of device heterogeneity on indoor localization accuracy, we conducted a comprehensive analysis using popular smartphones.

Wi-Fi RSS measurements reflect the power level of the wireless signal received from a Wi-Fi AP by a Wi-Fi receiver in devices such as smartphones, Tablets, and laptops. These measurements are expressed in decibels relative to a milliwatt (dBm) and typically range from -100dBm to 0dBm where -100dBm indicates no signal or invisible AP and 0dBm represents the strongest signal [85]. Weak signals are usually defined as below -80dBm, while strong signals are considered to be above -40dBm. However, it is important to note that the precise threshold values for strong and weak RSS signals can vary based on factors such as Wi-Fi chipset, device firmware, and the surrounding environment [85].

In practice, capturing RSS values in real-world settings is not as straightforward as in an idealized scenario. Multiple factors, such as signal attenuation, reflections from objects, human interferences, and multipath propagation, can introduce variations in the observed RSS values. Moreover, the presence of device heterogeneity further complicates the situation. Variations in hardware, software, and antenna configurations among different mobile devices, even with the same underlying Wi-Fi chipset (from the same manufacturer), can lead to significant discrepancies in the reported RSS values [85]. Firmware configurations and signal processing algorithms also contribute to differences in RSS readings. As a result, training ML models can be challenging as heterogeneous and noisy RSS data can result in poor generalization and inaccurate localization predictions.

To investigate the influence of device heterogeneity, we captured Wi-Fi RSS values using three different smartphones namely, OnePlus 3, Samsung Galaxy S7, and LG V20, at the same indoor

location and at the same time in a building. The observations are shown in Figure 23. Comparing the RSS values captured by each smartphone for various Wi-Fi APs, we can observe significant variations, that show the impact of device heterogeneity. For instance, when comparing the average RSS values for the AP with the first MAC ID ‘38:17:c3:1f:0e:10’, Samsung Galaxy S7 recorded -47 dBm, OnePlus 3 recorded -41 dBm, and LG V20 recorded -47 dBm (on average). Similarly, for the AP with the MAC ID ‘80:8d:b7:55:35:11’, Samsung Galaxy S7 captured -71 dBm, OnePlus 3 recorded -90 dBm, and LG V20 recorded -80 dBm (on average). Cumulative variations in hardware, firmware, and signal processing algorithms contribute to these differences and must be addressed in ML-based indoor localization systems to preserve accuracy across heterogeneous devices.

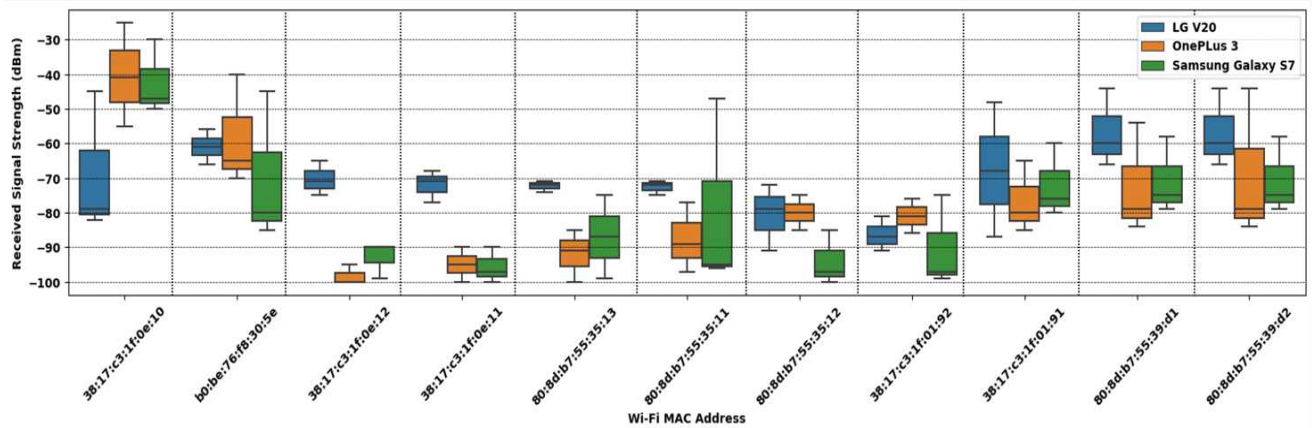


Figure 23: Analysis of device heterogeneity, across different smartphone at the same time and location

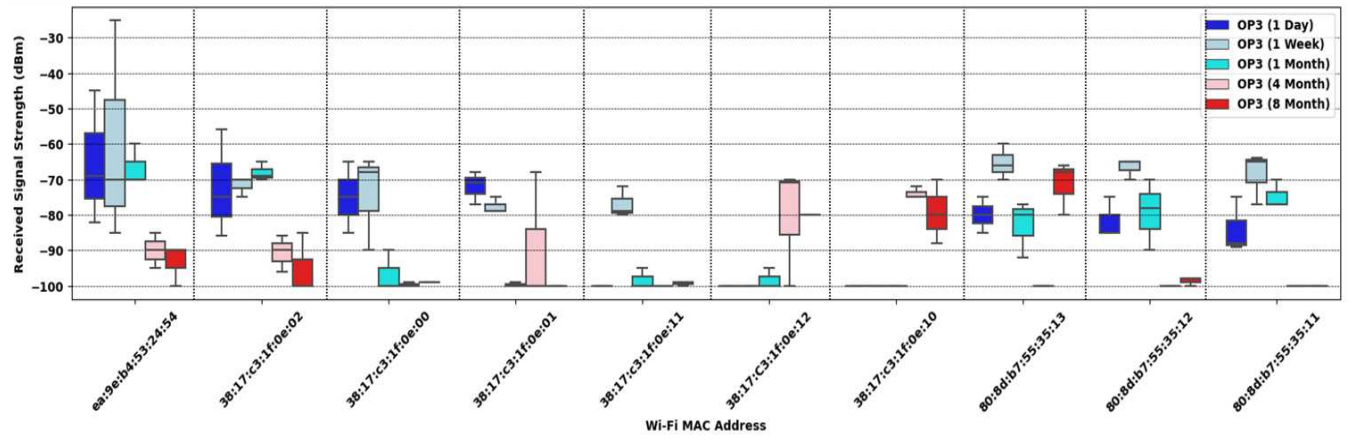


Figure 24: Analysis of temporal variations across varying time periods as captured by the same smartphone at the same location.

4.5. ANALYSIS OF TEMPORAL VARIATIONS

Temporal variations in indoor localization pose challenges due to changes in Wi-Fi APs or the indoor environment over time. Temporal variations can be attributed to several factors within the indoor environment, encompassing both short-term and long-term variations. Moreover, changes in AP configurations, such as APs being turned off, added, or dropped, lead to inconsistencies in RSS fingerprints and can impact the quality and availability of localization data, resulting in inaccurate predictions. Additionally, temporal variations can be categorized into short-term and long-term variations, each with its distinct impact on indoor localization. Short-term variations encompass rapid and transient changes within the indoor space. For instance, the movement of people, conversations in specific areas, or even adjustments like turning screens or opening metal cabinet doors can significantly influence local RSS values, introducing fluctuations in the data.

Long-term variations, on the other hand, involve more persistent alterations in the indoor environment. These can include structural changes like modifications in the floorplan or furniture layout, as well as the installation or removal of APs. Temporal variations play a crucial role in the real-world adoption of any indoor localization system, as they cause RSS fingerprints to undergo

significant changes over time, making indoor localization increasingly complex. To analyze the effects of temporal variations, we conducted Wi-Fi RSS data collection instances (CI) over several months at the same location. For each CI, we captured multiple fingerprints and subsequently averaged them to derive representative results, Figure 24 shows the results of the captured Wi-Fi RSS data with a OnePlus 3 smartphone. The data was collected during normal working hours to incorporate external noise, in addition to temporal variations caused by changes with the Wi-Fi APs.

One can observe significant fluctuations in RSS values for APs over the course of several months. For instance, on day 1, the average RSS value for the AP with MAC ID ‘38:17:c3:1f:0e:02’ was -75 dBm. However, at week 1 and at month 1 we observed changes for the RSS value for the AP to -70 dBm and -69 dBm (on average), respectively. Although these changes may not be considered significant, a more substantial variation occurred at months 4 and 8 when the RSS value dropped significantly to -90 dBm and -100 dBm (on average), respectively. These fluctuations in RSS values can be attributed to the impact of temporal variations, encompassing changes in AP configurations, both short-term and long-term variations. Such variations can have a detrimental impact on the performance of ML models used for localization, potentially resulting in mispredictions. We also noticed similar variations for other APs over the months. For example, considering the AP with MAC ID ‘38:17:c3:1f:0e:10’, we observed an average RSS value of -80 dBm at day 1. However, at week 1 and at month 1, the RSS value dramatically dropped to -100 dBm (on average), indicating that the AP was turned off during that period. Interestingly, at months 4 and 8, we observed a reversal of the previous state, as the AP became active again with RSS values of -70 dBm and -80 dBm (on average), respectively. These observations highlight the dynamic nature of indoor environments and the impact of temporal

variations on RSS readings. Addressing these challenges is crucial for developing a robust and long-term indoor localization solution.

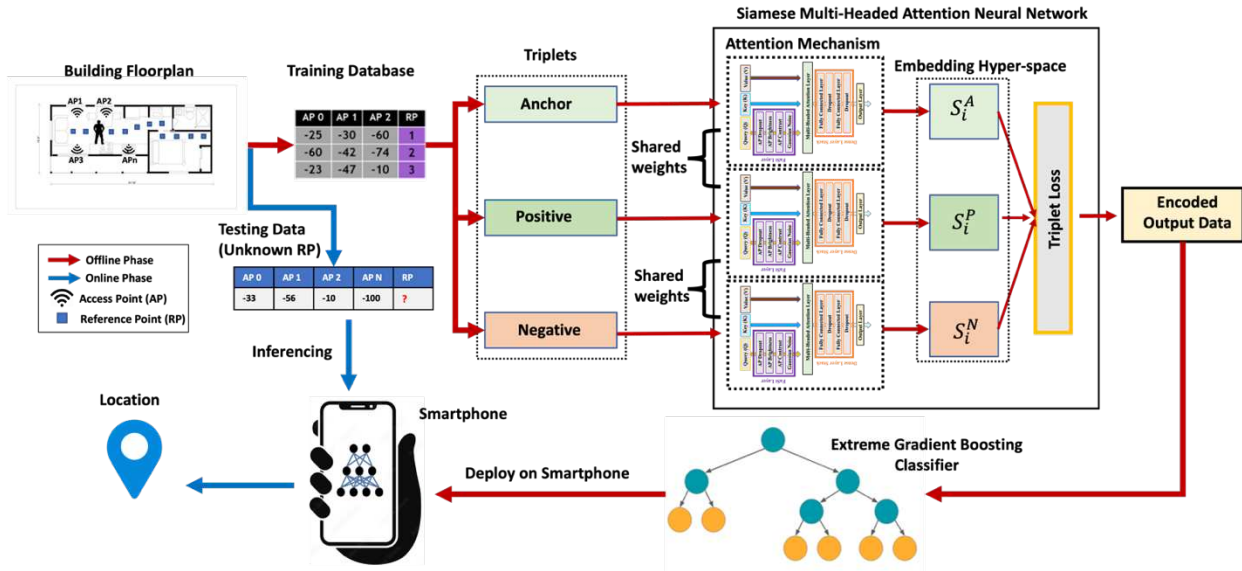


Figure 25: Working of the STELLAR framework, including offline (training) phase and online (inferencing) phase.

4.6. STELLAR FRAMEWORK

The STELLAR framework, an overview of which is shown in Figure 25, aims to address indoor localization challenges through a two-phase process. In the offline phase (red arrows), RSS fingerprints are collected from various reference points (RPs) across the building's floorplan. Each row in the training database represents the RSS values of visible APs for a specific RP, forming a single fingerprint. These fingerprints are then organized into three sets (triplets): anchor, positive, and negative, and sent to a Siamese multi-headed attention neural network model for training. The underlying attention model uses the inputs from each triplet to generate embedding hyper-spaces for each of the triplets, capturing characteristics related to device heterogeneity and temporal

variations. Details related to triplet selection and attention mechanism are discussed in the later sub-sections.

The triplet loss trains the multi-headed attention model’s output (the three embedding hyper-spaces), resulting in encoded versions of the triplets. The encoded data, along with associated RP locations, is then used to train an extreme gradient boosting (XgBoost) classifier. The XgBoost classifier improves performance and compensates for any negative impact from sub-optimal triplets. After this step, the offline phase concludes, and the trained ML models are now deployed on the smartphone for the online phase to predict location.

In the online phase, a user captures Wi-Fi RSS from different APs, using a smartphone at an unknown location. The RSS values are used to create a Wi-Fi fingerprint. This fingerprint is inferred by the Siamese model, generating an encoded output. The XgBoost classifier then processes this encoded data to predict the user's location.

The two-phase approach enables the STELLAR framework to address indoor localization complexities, incorporating device heterogeneity and temporal variations. Unlike most ML-based methods that typically require extensive training data collected from various locations, STELLAR is designed to deliver robust indoor localization with minimal training data. This efficiency is facilitated by the novel Siamese multi-headed attention neural network, combined with the proposed contrastive learning technique. These components empower STELLAR to achieve superior localization performance, even when faced with extreme RSS variations caused by temporal changes, all while using a limited amount of training data. In the following sub-sections, we describe the components of the STELLAR framework in more detail. In the following sub-sections, we describe the components of the STELLAR framework in more detail.

4.6.1. RSS FINGERPRINT PRE-PROCESSING

The database shown in Figure 25 contains RSS values for different Wi-Fi APs along with their corresponding RPs. The RSS values range from -100dBm to 0dBm, with -100 indicating no signal and 0 indicating the strongest signal, as discussed earlier. To prepare the training data for the Siamese multi-headed attention model (or inference data for prediction), the captured RSS values are normalized to a range of 0 to 1, where 0 represents weak or null signals, and 1 represents the strongest signal. Each fingerprint in this dataset is then used as a triplet input for the Siamese model. The details on selecting the three triplets are presented next.

4.6.2. TRIPLET SELECTION FOR CONTRASTIVE LEARNING

After pre-processing the fingerprints from the training database, they are divided into three triplets: anchor, positive, and negative, as shown in Figure 25. These triplets serve as inputs to train the Siamese multi-headed attention neural network. Each triplet is carefully selected to allow efficient training of the Siamese model.

The anchor consists of the captured fingerprint at a particular RP for training. The positive is formed by using the same fingerprint as the anchor, but with changes to a subset of RSS values. The positive is formed by randomly replacing a subset of the RSS values to -100dBm or 0 (normalized) to simulate changes in APs that can arise in the indoor space. For instance, AP:0 can be turned off due to maintenance or power outage and this can change the captured RSS fingerprint leading to inaccurate localization, simulating random AP dropout in the positive can prepare the model to address such variations. The negative is another fingerprint from the training database but located at a different RP from the anchor. For instance, if the anchor fingerprint is collected at RP:1, the negative fingerprint can be collected from an adjacent RP, such as RP:0 or RP:2.

Our Siamese model utilizes contrastive learning [86] during training to learn to identify similarities between the anchor and positive triplets, enabling it to adapt to scenarios where input RSS fingerprints are similar to the anchor but may have variations. Similarly, the model learns to differentiate between the anchor and negative triplets, allowing it to discern fingerprints from different RP locations, even when the anchor and negative triplets are close in locations. This powerful learning method finds similarities and dissimilarities between fingerprints, even when they exhibit low or high similarity characteristics, respectively. The Siamese model outputs an encoded representation of the triplets used for training.

4.6.3. ANCHOR TRIPLET SELECTION

The anchor represents the captured input fingerprint data used for training the Siamese model. It consists of RSS fingerprints captured at various RPs within the indoor environment, each having a unique set of visible Wi-Fi APs with their corresponding RSS values. The anchor plays a critical role as it serves as the baseline for training the network. It acts as a baseline against which the model learns to identify similarities and differences with other triplets (positive and negative). By using the anchor triplet as the basis for comparison, the Siamese model can effectively capture essential patterns and relationships within the data, enabling accurate localization predictions during the online phase.

4.6.4. POSITIVE TRIPLET SELECTION

The positive triplet is an essential part of the training process in the STELLAR framework. It is derived from the anchor triplet, which represents the input fingerprint data used to train the Siamese model. The positive triplet is created by randomly selecting a certain percentage ($D\%$) of

APs from the anchor triplet and setting their corresponding RSS values to -100 dBm (or 0 after normalization), effectively simulating the dropout of these APs. The introduction of dropout in the positive triplet serves the purpose of simulating the temporal variations that commonly occur in real-world indoor environments. By training the Siamese model using positive triplets with AP dropouts, the model learns to handle missing APs in the input data, improving its resilience to temporal variations.

4.6.5. NEGATIVE TRIPLET SELECTION

The purpose of introducing this negative triplet is to provide examples of fingerprints that closely resemble the anchor triplet, but at a different RP. This helps the Siamese model to learn to distinguish between similar yet distinct RPs and their corresponding fingerprints, enabling more accurate localization.

To find the negative triplet that shares the closest similarity with the anchor triplet, we use the Euclidean distance (ED) method, as shown in Equation 8. The ED method calculates the squared differences between the RSS values of the APs in the anchor and negative triplets, sums them, and takes the square root to obtain the final ED value. By measuring the overall differences between the triplets, the ED metric allows the model to evaluate their relative similarity or dissimilarity.

$$ED(F_{Anchor}, F_{Neg}[i]) = \sqrt{\sum (F_{Anchor} - F_{Neg}[i])^2} \quad (8)$$

Equation 8 represents the calculation of the ED between the anchor triplet (F_{Anchor}) and the negative triplet (F_{Neg}). The variable i refers to the RP location different from the anchor's RP. The selection process of the negative fingerprint (F_{Neg}) is shown in Equation 9.

$$F_{Neg} = \text{Argmin} (ED(F_{Anchor}, F_{Neg})). \quad (9)$$

The framework uses the *Argmin* operation to find the fingerprint with the minimum distance to the anchor triplet (F_{Anchor}). In other words, (F_{Neg}) (negative triplet) is chosen as the fingerprint from the training database that exhibits the closest similarity to the anchor fingerprint according to the ED metric. This process forces the Siamese model to learn dissimilarities between the similar triplets, as they are from different RPs.

4.6.6. SIAMESE MULTI-HEADED ATTENTION NEURAL NETWORK

Our Siamese model comprises of three multi-headed attention neural networks, each trained to accept one of the three triplets (anchor, positive, and negative) for training. These attention mechanisms share weights, enabling the model to learn common patterns and relationships across the triplets, enhancing its ability to handle challenges such as temporal variations and device heterogeneity effectively. During training, the multi-headed attention neural network generates embedding hyper-spaces for each triplet, capturing essential features and relationships within the fingerprint data (from the triplets), as shown in Figure 25. Utilizing the triplet loss with contrastive learning, the model learns to identify similarities between the anchor and positive triplets and distinguish between the anchor and negative triplet, providing a comprehensive understanding of the fingerprints' characteristics, as discussed in Section 4.6.2. The model's final output is an encoded representation of the three triplets used for training, containing valuable information about potential temporal characteristics and device heterogeneity. By leveraging the multi-headed attention neural network and the triplet loss with contrastive learning, the Siamese model provides

insights into the indoor environment's complexities, making it more resilient to temporal changes and heterogeneity, thereby leading to improved localization performance.

4.6.7. MULTI-HEADED ATTENTION NEURAL NETWORK

The multi-headed attention neural network within the Siamese model is designed with three inputs: Query (Q), Keys (K), and Values (V), as illustrated in Figure 26. Each of the three multi-headed attention neural networks handles one of the three triplets, with the triplet data fed into the Q input. Before passing through the Q input, the triplet data undergoes domain-specific augmentation using the FaSt (Fingerprint Augmentation Stack) module, discussed in Section 4.6.8. The K input represents the entire training database (fingerprints across all RPs), enabling comparisons with the Q input. On the other hand, the V input signifies the corresponding RP locations, of all the fingerprints in K in a one-hot encoded form. This one-hot encoding technique effectively represents categorical variables as binary vectors, where only one element is set to 1 while the rest are set to 0, enabling the network to process categorical information efficiently. This multi-headed attention mechanism allows the Siamese model to capture and focus on critical patterns and relationships within the fingerprint data for accurate indoor localization.

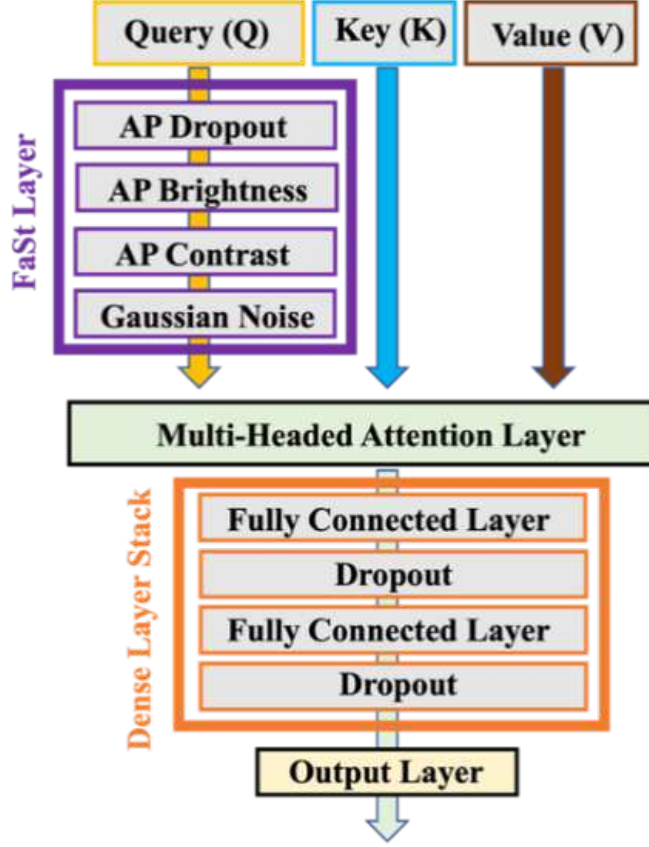


Figure 26: The algorithm of the Multi-headed attention mechanism.

The Q, K, and V are then fed into the multi-headed attention layer, as shown in Figure 26. The attention layer works by computing attention weights for each pair of Q and K elements and applying these weights to the corresponding V elements. The attention weights are calculated using the scaled dot-product attention mechanism, which measures the similarity between the Q and K elements. The attention layer allows the network to focus on relevant features in the fingerprint data and the corresponding RP locations, facilitating accurate indoor localization. The attention calculation is represented as follows:

$$Attention(Q, K, V) = softmax\left(Q K^T / \sqrt{d_k}\right) V \quad (10)$$

In the context of Wi-Fi RSS fingerprinting based indoor localization, Q represents the input fingerprint, K represents the fingerprint database (fingerprints across all RPs), and V represents the corresponding RP locations. The dot product between Q and K captures the similarity between the input fingerprint and the fingerprint database, while the *softmax* function normalizes the attention weights based on their relative strengths. The resulting attention weights are then applied to V to obtain the attended representation.

In multi-headed attention, this attention mechanism is applied multiple times with different linear projections of the Q, K, and V matrices. Let h represent the number of attention heads. The multi-headed attention can be mathematically represented as:

$$\begin{aligned} MultiHead(Q, K, V) &= Concat(h_1, h_2, \dots, h_n)W^o \\ \text{where } h_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \quad (11)$$

After the attention layer, the output is passed through a Dense Layer Stack, which consists of a series of fully connected layers with dropout regularization. The Dense Layer Stack effectively learns and extracts meaningful representations from the output of the attention layer, as shown in Figure 26. These layers enable the network to capture complex relationships and patterns in the data while preventing overfitting.

The multi-headed attention layer in the Siamese model uses shared weights across all heads to enable the layer to process all three triplets (anchor, positive, and negative) simultaneously. The resulting outputs for each triplet are stored as embedding hyper-spaces (i). These embedding hyper-spaces, representing the anchor (S_i^A), positive (S_i^P), and negative (S_i^N), are then trained

using triplet loss (discussed in Section 4.6.9). The combination of the attention mechanism and the Siamese method allows addressing the challenges posed by both heterogeneity and temporal variations during indoor localization.

4.6.8. FINGERPRINT AUGMENTATION STACK (FAST LAYER)

The FaSt module incorporates various powerful features aimed at enhancing the accuracy and robustness of indoor localization in real-world environments. One significant component is random AP dropout, wherein specific APs are randomly dropped from the RSS fingerprint data, followed by infilling the missing APs using Gaussian noise. This strategic approach can be effective in addressing device heterogeneity. As the model becomes more adept at handling diverse indoor environments with varying signal strengths.

In addition to random AP dropout and Gaussian noise, the FaSt module introduces two essential components: AP random brightness and AP random contrast. These features introduce variations in brightness and contrast levels, respectively, into the RSS fingerprint data. Brightness signifies the intensity or strength of the signal received from the APs, while contrast measures the difference between the highest and lowest signal strengths in the data. The module alters these signal strengths, by altering the intensities for AP random brightness and scaling the difference between the highest and lowest signal strengths for AP random contrast, effectively simulating different levels of signal intensity and contrast variations in the dataset. By incorporating these diverse patterns, the FaSt module equips the ML model with a comprehensive understanding of the potential variations that may be caused due to device heterogeneity.

4.6.9. TRIPLET LOSS

The triplet loss method plays a crucial role in the Siamese model, enabling it to learn similarities and dissimilarities between the triplets through contrastive learning. This loss function operates on the embedding hyper-spaces generated from the three triplets: anchor, positive, and negative. The relative distances between these embeddings in the hyper-space are used to optimize the overall model.

The main objective of minimizing the triplet loss is to ensure that the anchor and positive triplets, which represent similar fingerprints are close to each other in the hyper-space. At the same time, the anchor and negative triplets, which correspond to fingerprints from different RPs, should be pushed farther apart. This process facilitates the Siamese model in effectively identifying similarities between altered fingerprints (due to temporal variations or device heterogeneity) and distinguishing between fingerprints at different RPs, enhancing the accuracy and resilience of the indoor localization framework. By leveraging the triplet loss and contrastive learning, the Siamese model can create meaningful embeddings that are stored as the encoded output data. The mathematical equation for the triplet loss are:

$$F(Pos) = \text{Argmin} \left(||S_i^A - S_i^P|| \right)^2 \quad (12)$$

$$F(Neg) = \text{Argmax} \left(||S_i^A - S_i^N|| \right)^2 \quad (13)$$

$$\text{Triplet Loss} = \sum (F(Pos) - F(Neg)) \quad (14)$$

where S_i^A is the embedding hyper-space representation of the anchor triplet, S_i^P is the embedding hyper-space representation of the positive triplet, and S_i^N is the embedding hyper-space representation of the negative triplet. The triplet loss is used to output the encoded versions of the triplets.

4.6.10. EXTREME GRADIENT BOOSTING (XGBOOST)

The encoded output from the Siamese multi-headed attention neural network is fed into an Extreme Gradient Boosting (XgBoost) classifier for location prediction. As an ensemble learning algorithm, XgBoost learns from the encoded data and maps the learned features to their corresponding RP locations, enabling precise and robust indoor localization predictions. XgBoost combines the predictions of multiple weak models, called decision trees, to make a final prediction. Each decision tree is built iteratively, with each subsequent tree aiming to improve upon the previous ones. During training, the XgBoost algorithm computes a loss function (multi-class loss), which quantifies the difference between the predicted and target values. The algorithm then determines the optimal structure of the decision tree by minimizing the loss function.

Mathematically, the XgBoost algorithm aims to learn a prediction function $F(x)$ that maps the encoded data to the target location coordinates. The prediction function $F(x)$ can be represented as:

$$F(x) = W_0 + \sum W_j * F_j(x) \quad (15)$$

where W_0 is the bias term, W_j is the weight associated with the j -th decision tree, and $F_j(x)$ is the prediction made by the j -th decision tree for the input x . During training, the XgBoost algorithm

optimizes the weights (W_j) and bias term (W_0) by minimizing a combination of the loss function. This process ensures that the decision trees collectively capture the complex relationships between the encoded input data and the target location coordinates.

4.7. EXPERIMENTS

In this section, we describe the experimental setup to assess the performance of our proposed STELLAR framework in real-world scenarios and compare it with state-of-the-art indoor localization frameworks, including: the multi-headed attention framework (ANVIL) [45], the vision-transformer framework (VITAL) [80], the autoencoder support gaussian process classifier framework (WiDeep) [17], the stacked autoencoder with gradient boosting framework (SANGRIA) [82], and the long-term K-nearest neighbor framework (LT-KNN) [83]. Our experiments test the frameworks for resilience towards both device heterogeneity and temporal variations. To ensure resilience of all frameworks towards these challenges, we train all the frameworks using data collected from a single smartphone across multiple RPs in the building floorplan at the same collection instances (CI) or time period. This training enables the frameworks to be evaluated on various smartphones over an extended period without requiring re-training.

The training process of the STELLAR framework begins by setting the $D\%$ of APs (in the negative triplet) to be dropped to 60% (see Section 4.6.5) and then training the Siamese multi-headed attention neural network with 7 heads and a head size of 50, where each head represents an individual sub-network module responsible for processing input features and head size specifies the dimensionality of each head (see Section 4.6.7). The FaSt module, consisting of AP dropout (0.1), AP random contrast (0.1), AP random brightness (0.1), and Gaussian noise (0.12), all designed to introduce variability that can arise due to heterogeneity and enhance the robustness

and adaptability of the model (see Section 4.6.8). The Dense Stack Layer, comprising fully connected layers with ReLU (rectified linear unit) nonlinearity, has 128 and 64 neurons, respectively, while the output layer uses the Softmax function (see Section 4.6.7). During training, the Siamese multi-headed attention neural network is optimized with a learning rate of 0.0001 for 300 epochs and is trained using the triplet loss (see Section 4.6.9), resulting in a total of 201,672 trainable parameters. The XgBoost classifier is set with a maximum depth of 7 (see Section 4.6.10).

To evaluate the performance of a localization framework, we measure the localization error using the Euclidean distance (ED), which quantifies the dissimilarity between the predicted and actual (target) location. This metric provides valuable insights into the effectiveness of a framework in accurately estimating positions in indoor environments. The localization error is calculated based on the distance between the ground truth location and the estimated location of a device, which is based on the RSS data collected by the device. This approach allows us to assess the localization error of the framework across multiple devices, building scenarios, and time periods, ensuring that the framework can be reliably used in real-world settings.

4.7.1. DATA COLLECTION AND PRE-PROCESSING

To gather information for our indoor localization experiments, we utilized six smartphones from different manufacturers, each equipped with a distinct Wi-Fi chipset, as shown in Table 4.

Table 4: Details of smartphones used in our experiments.

Manufacturer	Model	Acronym	Wi-Fi Chipset
BLU	Vivo 8	BLU	MediaTek Helio P10
HTC	U11	HTC	Qualcomm Snapdragon 835
Samsung	Galaxy S7	S7	Qualcomm Snapdragon 820
LG	V20	LG	Qualcomm Snapdragon 820
Motorola	Z2	MOTO	Qualcomm Snapdragon 835
OnePlus	3	OP3	Qualcomm Snapdragon 820

It is worth noting that although some of the Wi-Fi chipsets used are the same, each smartphone manufacturer implements different firmware for noise filtering, resulting in inherent heterogeneity even when the Wi-Fi chipsets are identical. We selected these devices to capture both heterogeneity due to hardware and software.

We conducted experiments in two distinct buildings, each with unique characteristics, to evaluate the indoor localization frameworks, as shown in Figure 27. Building 1 featured a combination of wood and cement materials, spacious areas with computers, and a total of 160 identifiable Wi-Fi APs. Building 2 had a modern design with a mix of metal and wooden structures, open spaces, and shelves equipped with heavy metallic hardware. We identified 218 unique APs in Building 2. These buildings were located close to each other, separated by a single wall, enabling us to study the impact of interference from different sources across the buildings.



The complex indoor environment, coupled with the presence of metal and electronic devices near the test paths, resulted in noisy Wi-Fi fingerprints, posing challenges for accurate indoor localization. To capture realistic scenarios, we collected Wi-Fi RSS fingerprints during regular working hours, without artificially controlling human occupancy. We collected a total of 6 fingerprints per RP at each collection instance (CI), spanning a duration from 30 seconds to several months. We captured 16 CIs (CIs:0-15) over a total span of 8 months. To capture the impact of varying human activity throughout the day, we performed the first three CIs (0-2) for both paths on the same day, with a time interval of 6 hours between each CI. This allowed us to capture fingerprints early in the morning (8 A.M.), at mid-day (3 P.M.), and late at night (9 P.M). Subsequently, we performed CIs 3 to 8 across 6 consecutive days. The remaining CIs (9-15) were carried out at approximately 30-day intervals.

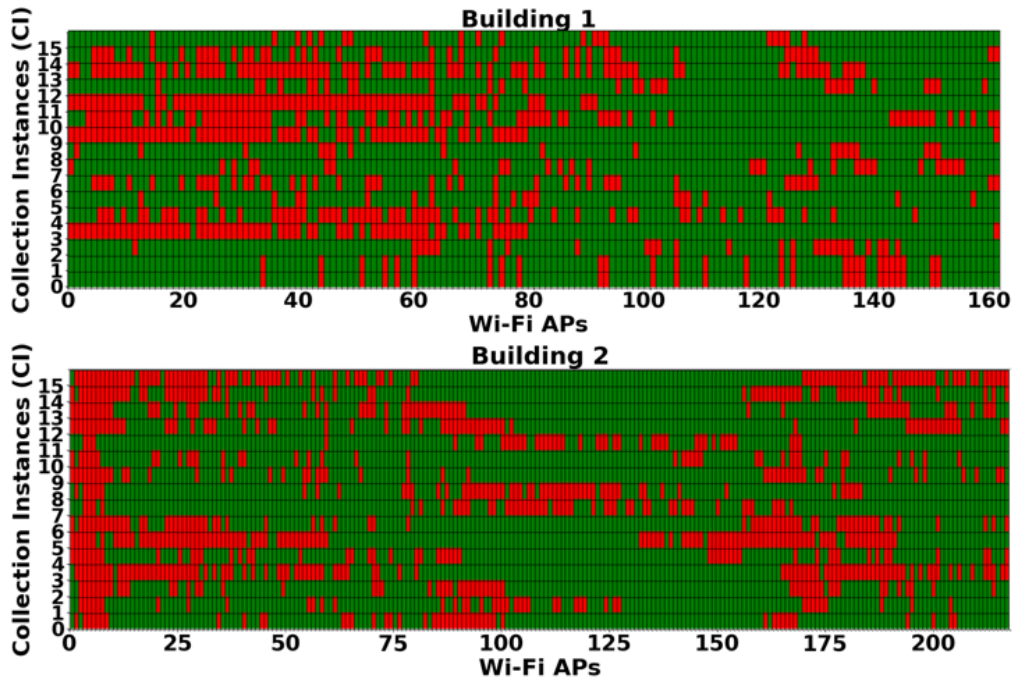


Figure 28: Analyzing visible (green) APs and invisible (red) APs over time.

Figure 28 illustrates the ephemerality of Wi-Fi APs along the Building 1 and Building 2 paths across the 16 CIs (CIs:0-15) over the total duration of 8 months. Each CI is represented on the y-axis, while the x-axis displays the specific Wi-Fi APs. Red marks indicate that a particular AP was not observed during the corresponding CI and green marks indicate visibility of the AP. We did not observe any significant changes in AP visibility up to CI:2. However, between CI:3 and CI:9, approximately 20% of the Wi-Fi APs became unavailable. After CI:10 approximately 40% of the Wi-Fi APs became unavailable. For the Building 1 and Building 2 paths, we specifically utilized a subset of CI:0, capturing fingerprints during the early morning, for the offline phase of our experiments. This means that the training process solely relied on this subset of data from CI:0. The remaining data from CI:0 and CIs:1-15 were reserved for testing purposes.

By collecting Wi-Fi RSS fingerprints over a duration of seconds to months, including customary working hours, we were able to capture various forms of interference and obstacles that impact Wi-Fi signal propagation. This approach allowed us to account for dynamic factors such as population density and human movement within the building, providing a realistic understanding of the challenges encountered in real-world indoor localization scenarios. In our dataset, we split the data in a 5:1 ratio to ensure a robust evaluation of the model's performance. During the training phase, we utilized 5 fingerprints per RP, while for the online phase, we used 1 fingerprint per RP. We employed a granularity of 1-meter separation between RPs, which we deemed sufficient for accurate localization of individuals. Additionally, we have made our data openly available [59] for reproducibility and to support further research in the indoor localization community.

4.8. EXPERIMENTAL RESULTS

In this section, we present the results of various experiments conducted to assess the performance of our STELLAR framework. We first present the outcomes of hyperparameter tuning to optimize the STELLAR framework. Subsequently, we conduct an ablation study to explore the selection process of the gradient boosting classifier. Next, we compare the STELLAR framework with state-of-the-art indoor localization frameworks to evaluate its effectiveness in handling device heterogeneity. Additionally, we compare the STELLAR framework with existing approaches to assess its performance under temporal variations. Lastly, we conclude by subjecting all frameworks to an extended evaluation, considering newer diverse devices (different devices from Table 4) and extreme temporal variations to ascertain their robustness under extremely challenging conditions.

4.8.1. HYPERPARAMETER TUNING: AP DROPOUT

In this experiment, we investigate the impact of dropping a percentage (D%) of APs (for the negative triplet, see section 4.6.5) during the training of the Siamese multi-headed attention neural network in the STELLAR framework. In this experiment, we investigate the impact of dropping a percentage (D%) of APs (for the negative triplet, see section 4.6.5) during the training of the Siamese multi-headed attention neural network in the STELLAR framework. We randomly drop D% of APs to introduce variability, enabling the model to effectively adapt to variations arising from temporal changes. This randomization simulates scenarios where APs may be missing or turned off without following a predefined pattern. The purpose of this experiment is to analyze how the framework performs when dealing with scenarios where certain APs may be missing or

turned off. For the experiment, we selected the OP3 device as the training device and used a subset of data from CI:0 for training the STELLAR framework.

The performance of the framework was evaluated on the remaining devices across all CI values (ranging from 0 to 15) across both the building floorplans. The localization error was reported in meters and calculated using ED, without re-training on any of the CIs (re-calibration-free). We varied the value of $D\%$ (at the negative triplet, see section 4.6.5) from 10% to 90% in steps of 10%, representing the percentage of APs that are dropped during training. The dropped APs were simulated by setting their corresponding RSS values to -100 dBm (or 0 after normalization). The average localization error per value of $D\%$ was computed by considering the localization errors from all testing devices and CIs. These results are shown in Figure 29.

From the results, we can observe that as the percentage of dropped APs increases from 10% to 60%, the average localization error decreases gradually. This suggests that the framework can adapt and compensate for the missing APs. However, beyond 60% of dropped APs, the average localization error starts to increase. This behavior may occur because a higher percentage of dropped APs introduces a greater loss of information, making it more challenging for the framework to accurately estimate the device's location. Thus, highlighting the importance of selecting an appropriate value of D to ensure accurate indoor localization. By identifying the optimal value of D , we can enhance the framework's performance and ensure its robustness in real-world scenarios. In this study, we use the specific value of D (60%) that demonstrated the best performance for all the subsequent experiments.

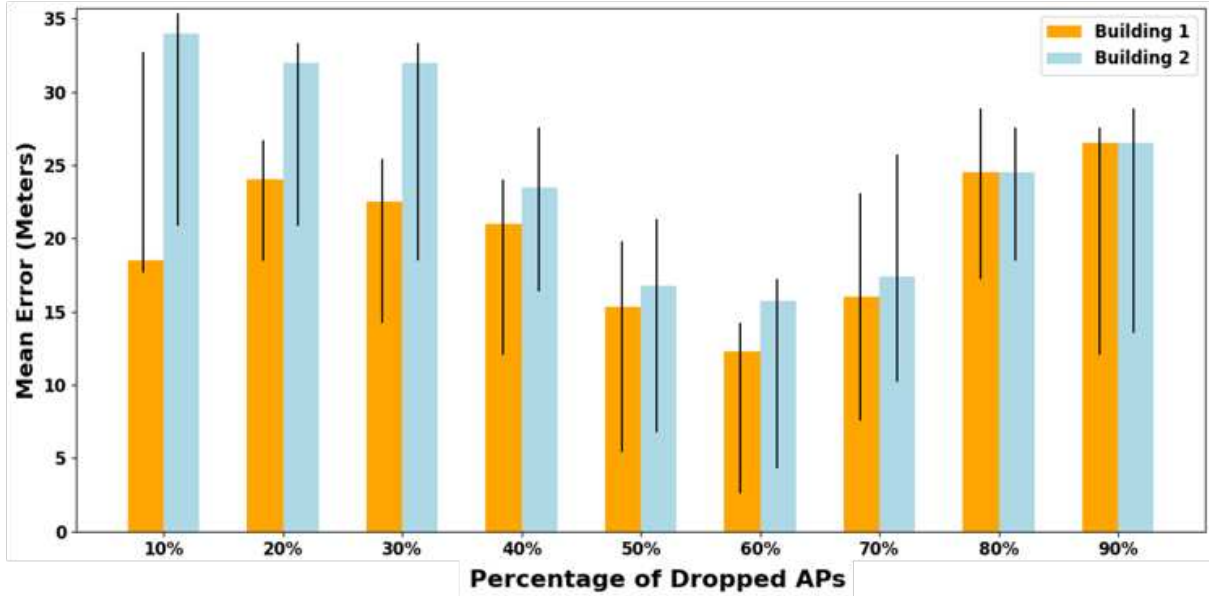


Figure 29: Tuning D% of AP dropout during offline phase for both building floorplans with min-max intervals showing best and worst-case errors for each D% value.

4.8.2. ABLATION STUDY: POST ENCODING CLASSIFIERS

In this ablation study, we explore the performance of different non-parametric models when training the output from the Siamese multi-headed attention neural network (encoded output). The preference for non-parametric models arises from their capability to handle the complexities of the encoded output without the constraints of fixed parameters, resulting in more flexible and adaptive solutions. The non-parametric model is essential to prevent suboptimal triplet selection which can hinder learning and negatively impact the model's performance (see Section 4.6.9). The goal is to compare the accuracies achieved by popular non-parametric algorithms: KNN (k-nearest neighbors) where the number of neighbors is set to 4 for determining the nearest data points for prediction, RF (Random Forest) with 100 estimators for creating an ensemble of decision trees, SVM (Support Vector Machine) with the radial basis function (RBF) as a kernel to transform the

data and find optimal decision boundaries, CatBoost (categorical gradient boosting) with a depth of tree set to 7, and XgBoost (extreme gradient boosting) with a depth of tree to 7, all chosen based on the values that provided the best results during the exploration process.

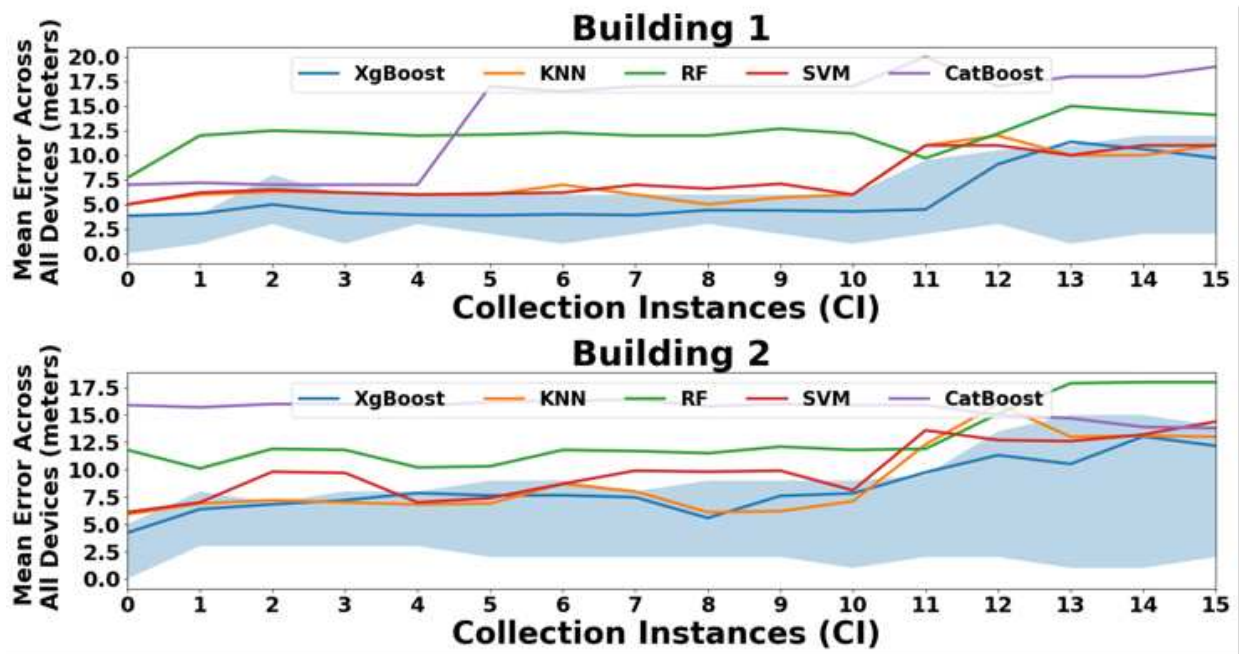


Figure 30: Comparing response of different classifier in temporal space for both building floorplans. Blue shadow indicates best and worst-case errors for STELLAR.

We selected the OP3 device as the training device and utilize a subset of data from CI:0 to train all frameworks. The performance of the framework is then evaluated on the remaining devices (from Table 4 across all CI values (ranging from 0 to 15) across both the building floorplans. The localization errors, reported in meters, are averaged across all devices for each CI, and are shown in Figure 30.

The KNN, SVM and XgBoost models consistently outperform the other models, particularly at CI greater than 9 where a significant number of APs are dropped. The KNN model excels in capturing local patterns and finding nearest neighbors, resulting in low localization errors. The

SVM model performs well due to its ability to create effective decision boundaries and handle complex data distributions. The XgBoost excels as the most effective model, employing a powerful ensemble of decision trees to capture intricate data relationships, robustly handling complexities from device heterogeneity and temporal variations, resulting in superior indoor localization accuracy. RF exhibits stable performance but is less adept at handling the dropped AP scenario, as it relies on averaging the predictions of multiple decision trees, and each tree is constructed independently using a random subset of features and data points. CatBoost, a gradient boosting algorithm designed to handle categorical features, may perform poorly in scenarios involving dropped APs, exhibiting similar limitations to the RF algorithm. As a result, it may not be optimal for addressing the challenges presented in this study. In conclusion, based on the analysis, XgBoost proves to be the top performer demonstrating the lowest localization error, adaptability to handle heterogeneity, and temporal variations, making it the ideal choice for subsequent experiments in long-term indoor localization.

4.8.3. ABLATION STUDY: SENSITIVITY OF TRAINING DATA

In this ablation study, we delve into the sensitivity of the STELLAR framework to variations in the training data. Our objective is to assess the framework's performance across different smartphones and building floorplans by adjusting the number of fingerprint samples collected per RP, during the training phase. We explore various training scenarios, ranging from utilizing 1 to 5 fingerprint samples per RP, as shown in Figure 31. The X-axis represents all CIs over a total span of 8 months, while the Y-axis denotes the mean localization errors (averaged across all smartphones and RPs within the respective building floorplans). Additionally, we include a shaded region for the best performing sample to display the variability of localization performance (with

the region spanning between the best and worst-case scenarios). We selected OP3 as the training device from CI:0 for all training samples, , this choice was made to demonstrate the robustness of the STELLAR framework, even when trained with a single or limited training device. This choice underscores the independence of the STELLAR framework from the choice of the training device. Further details can be found in Section 4.8.5, which depicts the minimal impact on STELLAR's performance when utilizing different training devices. The designation CI:0 indicates that the training data used was collected only from the first instance.

Our findings indicate that the fingerprint samples collected per RP during the training phase impacts the efficacy of STELLAR. Notably, employing just one fingerprint per RP results in the poorest performance. This can be attributed to the inadequacy of the training data to account for the multifaceted challenges posed by indoor environments, such as device heterogeneity, temporal variations, and the presence of external noise. Conversely, as the number of fingerprint samples per RP increases from two to five, STELLAR's performance exhibits a notable and consistent improvement. This is reflected in the reduction of localization errors across both temporal and heterogeneity spaces.

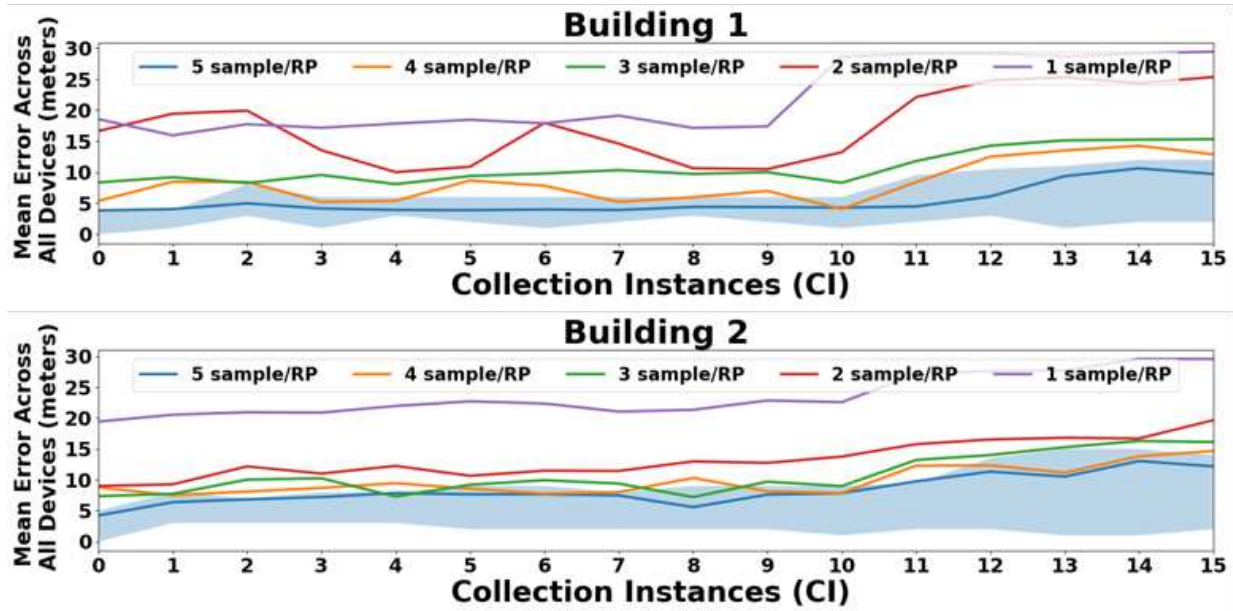


Figure 31: Sensitivity analysis of training data on STELLAR's performance. Blue shadow indicates best and worst-case errors for STELLAR.

4.8.4. COMPARISON VS. STATE-OF-THE-ART (HETEROGENEITY)

In this experiment, we evaluate the performance of the STELLAR framework in comparison to state-of-the-art localization frameworks, with a specific focus on device heterogeneity. To ensure transparency and reproducibility, we specifically selected the state-of-the-art localization frameworks with open-source implementations. To assess the impact of different training devices, we train each framework using data from a single device and evaluate its accuracy on the remaining testing devices. By observing any significant changes in performance across different training devices, we can determine the framework's resilience towards heterogeneity.

Figure 32 presents the results of this analysis, showcasing the performance of all frameworks in a matrix format while Figure 33 summarizes the average errors. In Figure 32, the x-axis represents the training devices, while the y-axis represents different frameworks, and the z-axis

represents the average localization errors (meters) across all the testing devices. The experiments were performed on both the floorplans (building 1 and 2). We specifically selected the CI:0 data from both training and testing devices in the building 1 and building 2 floorplans. All frameworks aim to address device heterogeneity except LT-KNN which is designed to address temporal variations.

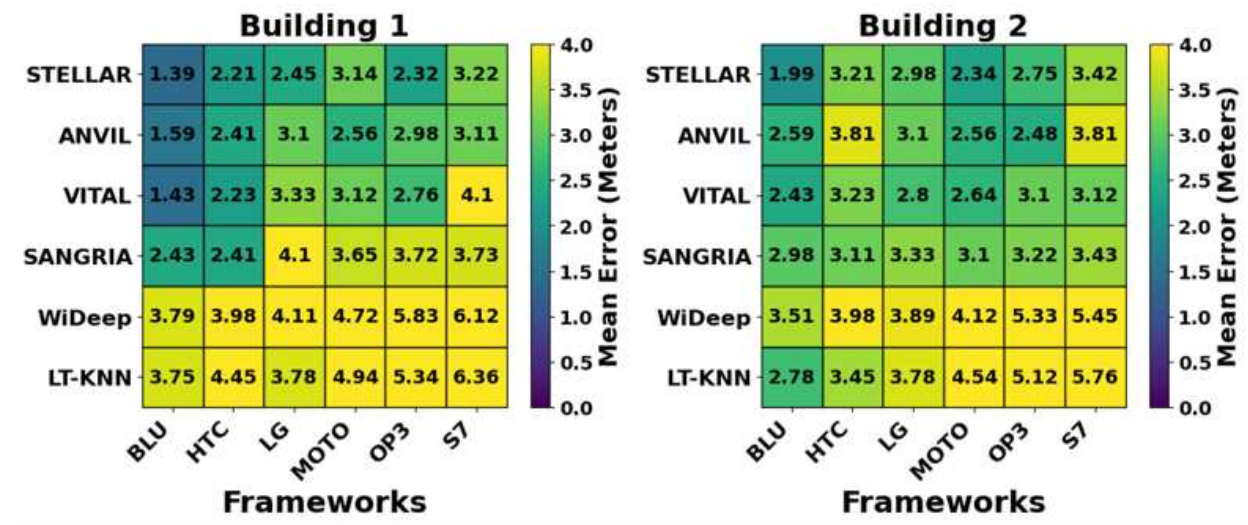


Figure 32: Framework comparison: average localization errors (meters) across devices.

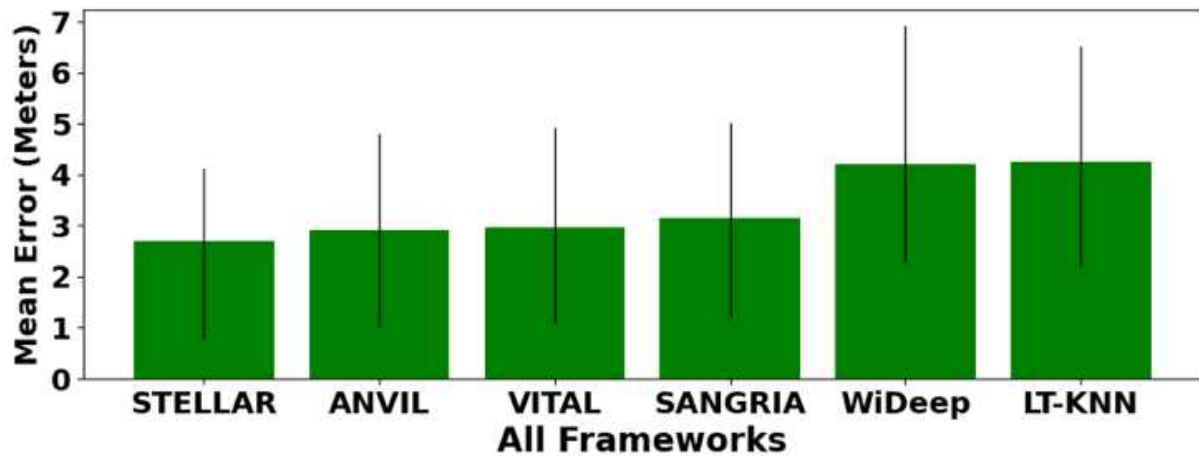


Figure 33: Average errors across training - testing devices from both building floorplans with min-max intervals showing best and worst-case errors for each framework.

As an example of how to interpret Figure 32, the first column for both buildings are the case where the BLU device was used as a training device, and the numbers in the column are the average localization error across the testing devices (remaining 5 smartphones from Table 4). It can be observed that ANVIL performs well due to its attention-based approach that captures relevant spatial dependencies. VITAL also excels with its vision transformer-based model, effectively modeling complex relationships. SANGRIA's stacked autoencoder architecture enables robust handling of device heterogeneity. STELLAR reports the lowest localization errors compared to its competitors. From Figure 33, STELLAR shows improvements over ANVIL (by 8.81%), VITAL (by 9.19%), SANGRIA (by 24.9%), WiDeep (by 72.41%), and LT-KNN (by 75%) in addressing device heterogeneity.

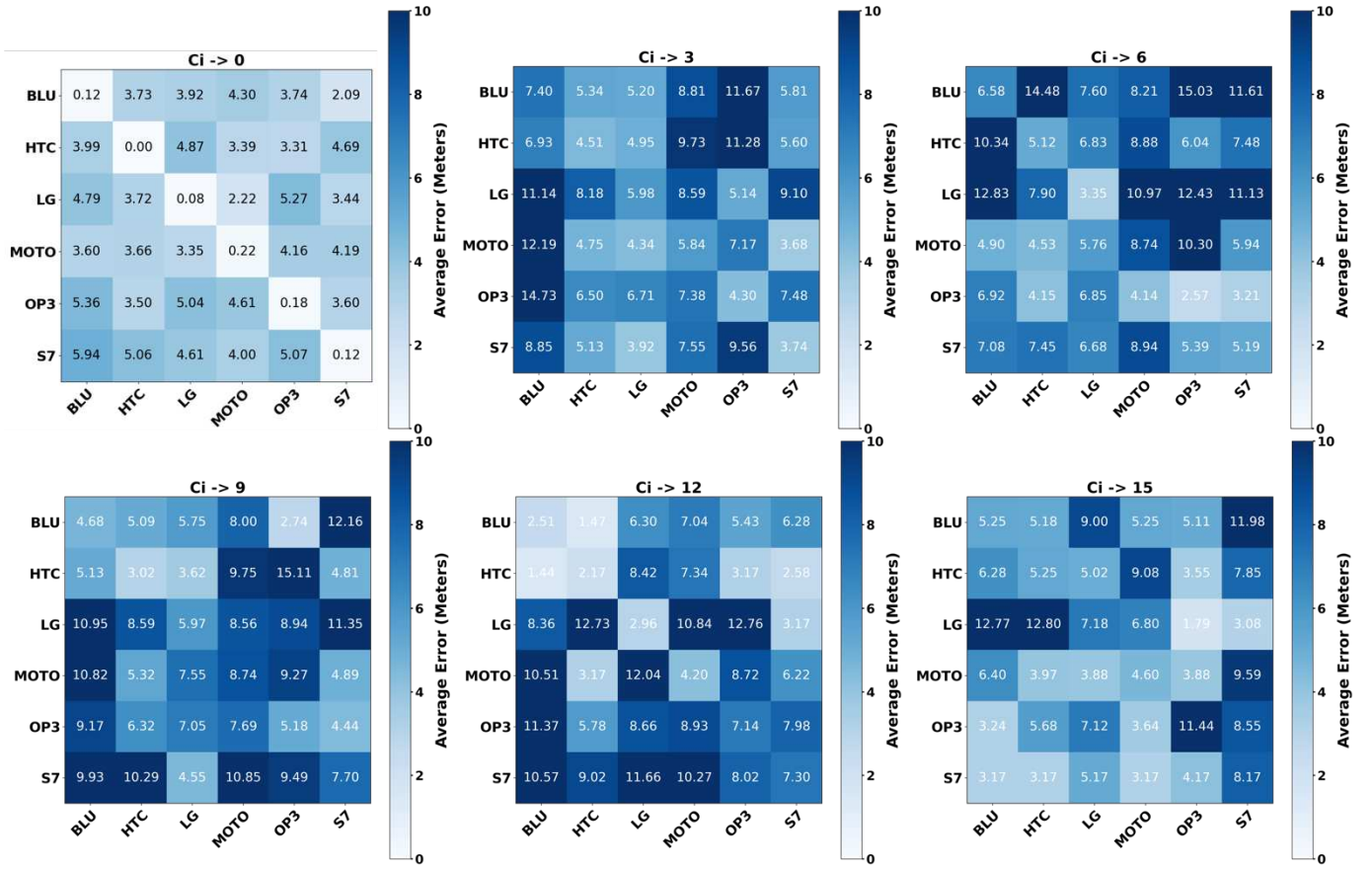


Figure 34: STELLAR’s performance analysis: average localization error (meters) across training and testing devices with varied temporal variations (CIs)

4.8.5. COMPARISON VS. STATE-OF-THE-ART (TEMPORAL VARIATIONS)

In this study, we analyze the performance of the STELLAR framework in terms of temporal variations and compare it with state-of-the-art localization frameworks. We also conduct an in-depth analysis of the framework’s ability to address heterogeneity in the temporal space. Figure 34 illustrates the performance of STELLAR across all training and testing devices, and across six different CIs. The x-axis represents the testing device, the y-axis represents the training device, and the z-axis represents the average localization error in meters. We focus on specific CIs: CI:0 (day 1), CI:3 (1 week), CI:6 (1 month), CI:9 (2 months), CI:12 (5 months), and CI:15 (8 months)

to explore performance with temporal variations across different timespans. For brevity, we omit the results of other CIs, as they exhibit similar patterns to those depicted in Figure 34.

To further assess STELLAR's performance across different CIs, we present Figure 35, which displays the variance between the best and worst-performing testing devices for specific training devices and CIs. The x-axis represents the training device, the y-axis denotes the CIs, and the z-axis indicates the difference in performance between the best and worst testing devices. Figure 35 analysis reveals that, for the majority of cases, STELLAR exhibits consistent error rates, even in the presence of temporal variations. This indicates the ability of the STELLAR framework to address the challenges of heterogeneity, primarily attributed to its attention mechanism that effectively handles the fluctuating nature of RSS values. Furthermore, we note that STELLAR maintains strong performance even at later CIs (greater than CI:9), demonstrating its ability to handle temporal variations. This robustness can be attributed to the contrastive learning, which effectively captures variations in the temporal space and enables STELLAR to serve as a long-term, resilient localization framework.

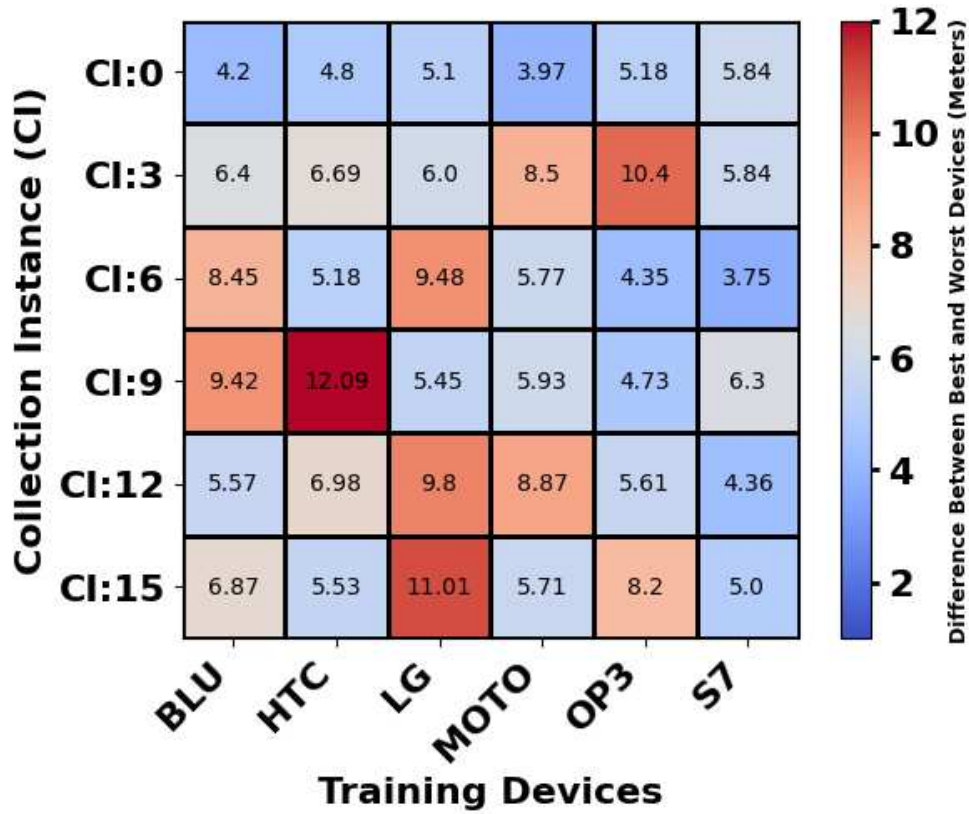


Figure 35: Difference between best and worst performing testing device per training device per CI.

To gain deeper insights into the performance of the STELLAR framework, we present Figure 36, showcasing the average localization errors across different CIs in a line plot. This analysis includes a comparison with state-of-the-art localization frameworks, focusing on the responses of each framework in different building floorplans, capturing both temporal and heterogeneity aspects. From Figure 36, it becomes evident that STELLAR consistently achieves lower localization errors compared to other frameworks, in both building 1 and building 2. Notably, the response from STELLAR remains relatively stable, highlighting its ability to maintain accurate localization despite changes in the environment.

ANVIL and VITAL, although designed to address heterogeneity, show some resilience in the temporal space. However, they tend to exhibit higher localization errors, particularly when confronted with a large number of dropped APs (CI:9 and above). On the other hand, SANGRIA and WiDeep, also designed for heterogeneity, are significantly influenced by temporal variations. These frameworks struggle to maintain accuracy when confronted with changing environmental conditions over time. It is worth noting that the LT-KNN framework requires frequent re-training (after every third CI) and still fails to outperform STELLAR. This is because frequent re-calibrations (i.e., re-training) can potentially lead to overfitting in most ML models.

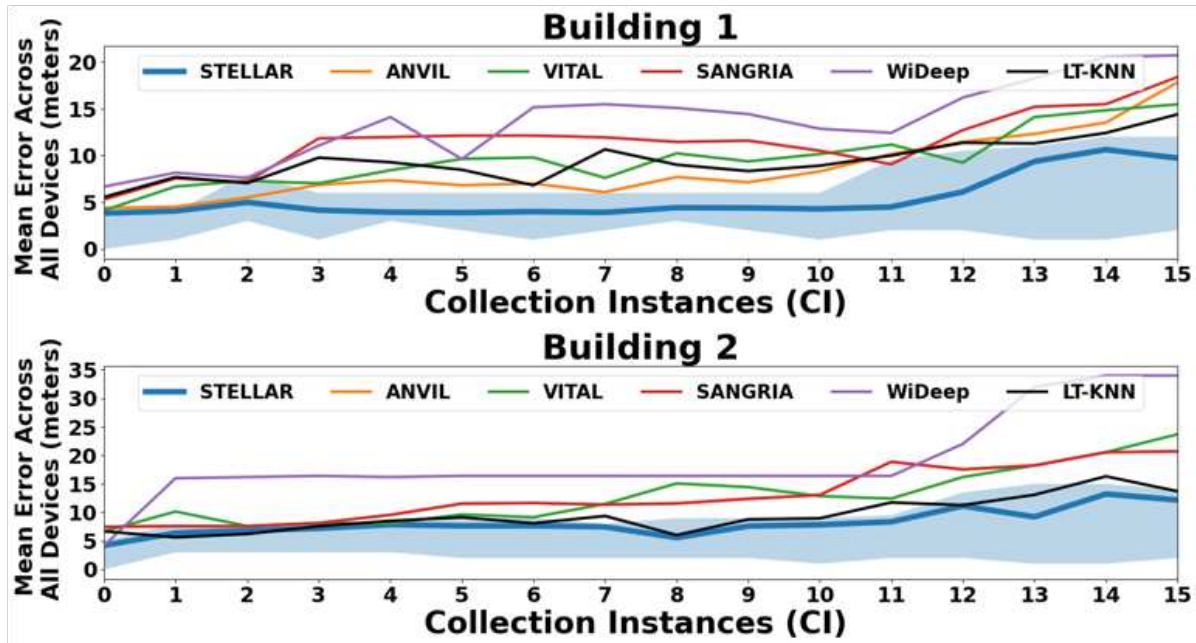


Figure 36: Average errors across all frameworks in the temporal space, blue shadow indicates best and worst-case errors for STELLAR.

To better visualize the performance disparity between STELLAR and the other frameworks, Figure 37 illustrates a series of box plots, each corresponding to a unique CI. The box plots depict the discrepancy in errors between STELLAR and its competitors for each CI. The lower whisker

of the box plot denotes the smallest error difference found when comparing STELLAR with the competing frameworks, while the upper whisker indicates the highest error difference observed. The central black bar within each box represents the mean error difference. A closer examination of Figure 37 reveals an upward trend in the error difference as the CIs increase. This suggests that as temporal variations become more pronounced over time, the competing frameworks struggle to sustain uniform performance, which highlights the relative robustness of STELLAR.

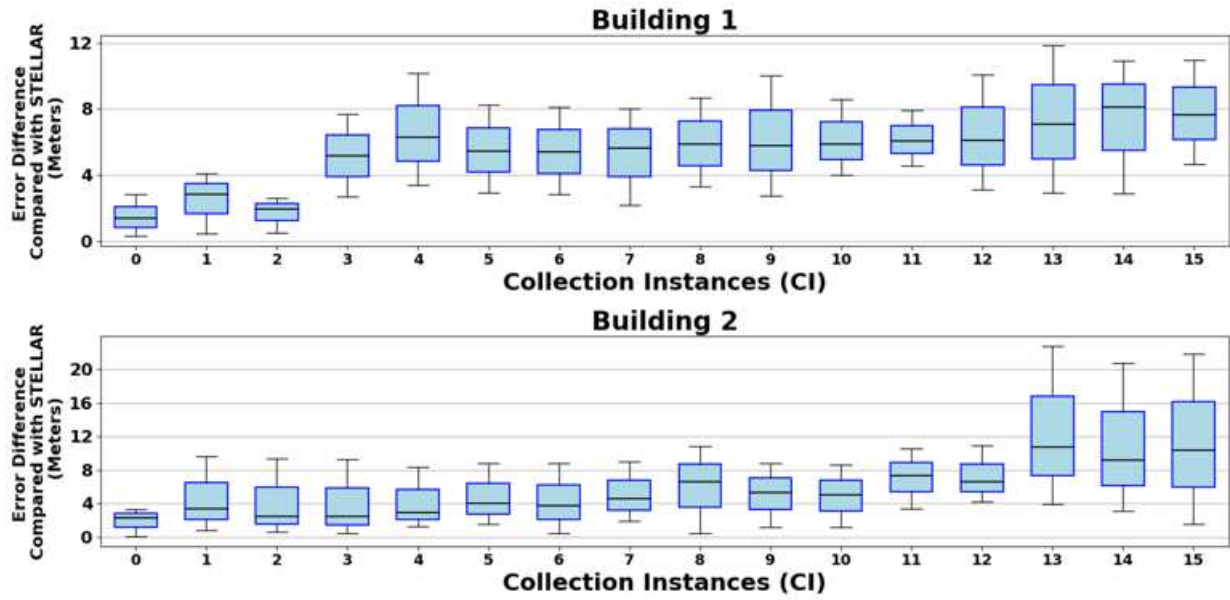


Figure 37: Box plot illustrating the minimum, maximum, and mean error differences between STELLAR and other frameworks per CI.

In conclusion, the results highlight the superior performance of the STELLAR framework in handling both heterogeneity and temporal variations. Comparing its performance with other frameworks in building 1, STELLAR demonstrates significant improvements of 75.2% over ANVIL, 80.11% over VITAL, 114.12% over SANGRIA, 153.34% over WiDeep, and 58.92% over LT-KNN. Similarly, in building 2, STELLAR outperforms ANVIL by 62.65%, VITAL by 59.89%, SANGRIA by 73.68%, WiDeep by 139.04%, and LT-KNN by 18.29%.

4.8.6. EVALUATION ON EXTENDED DEVICES AND TIME PERIOD

To enhance our understanding of the STELLAR framework’s resilience in handling heterogeneity and temporal variations, we conducted an extended evaluation using additional data from newer smartphone devices. In this experiment, we evaluate the performance of all the frameworks on fingerprint data collected, only by the newer smartphone devices presented in Table 5. The data was collected on the same building floorplans (building 1 and building 2) with a significant time gap of 16 months from the previously collected CI (CI:15). This temporal difference of 16 months effectively spans a period of 2 years (from CI:0), allowing us to capture and analyze data from time intervals ranging from a few seconds to 2 years. The newly collected CI, now denoted as CI:16, provides valuable insights into the evolution of the indoor environment. Notably, we observed a staggering 60% change in the APs within this time frame, representing a substantial alteration in the wireless infrastructure. To further intensify the testing and explore extended heterogeneity challenges, we incorporated newer testing devices into the evaluation. The details of these extended devices of expanded study can be found in Table 5.

Table 5: Details of extended smartphones used in our experiments.

Manufacturer	Model	Acronym	Wi-Fi Chipset
Apple	iPhone 12	iPhone	Qualcomm Snapdragon X55
Google	Pixel 4a	Pixel	Qualcomm Snapdragon 730G
Nokia	7.1	Nokia	Qualcomm Snapdragon 836

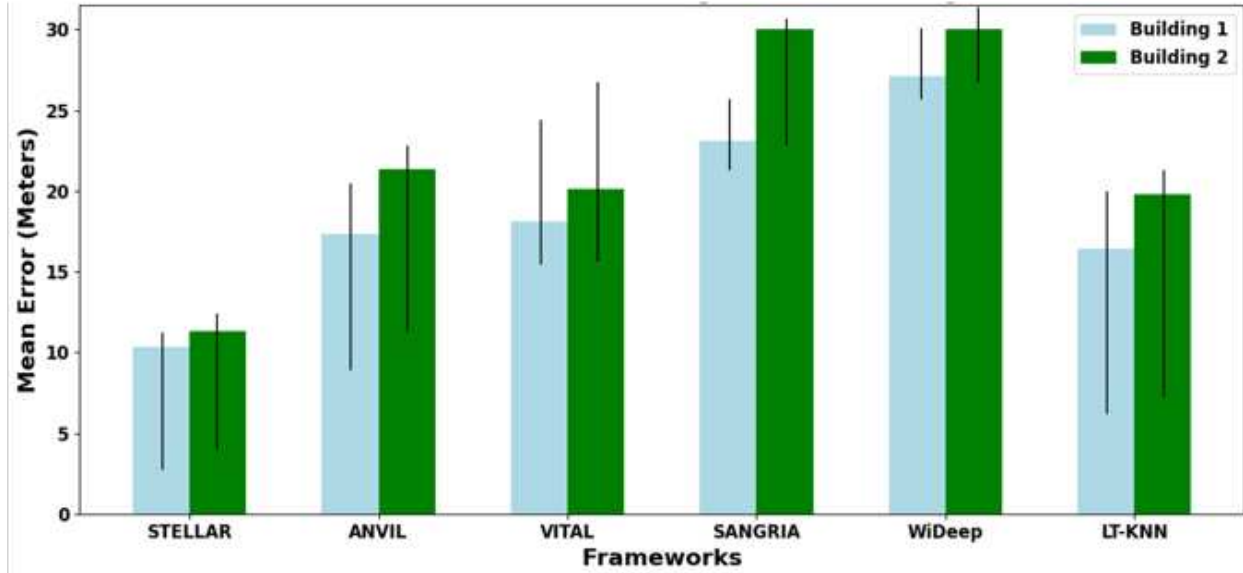


Figure 38: Average errors across all frameworks in the temporal space for extended devices with min-max intervals showing best and worst-case errors for each framework.

Figure 38 shows the performance of various frameworks on the extended devices and for CI:16. It is evident that, compared to STELLAR, all other frameworks exhibit poor performance as the intensity of temporal and heterogeneity increases, resulting in degraded performance. It is worth noting that the LT-KNN framework requires frequent re-training and, in this experiment, we re-trained LT-KNN on the previous CI (CI:15) and it still fails to outperform STELLAR. This may be because frequent re-training can potentially overfit some ML models. Additionally, it can be observed that the average error for STELLAR has also increased compared to the previous CI:15. However, it remains significantly lower than the errors produced by the state-of-the-art frameworks being compared. In building 1, STELLAR improves upon ANVIL by 68%, VITAL by 75.5%, SANGRIA by 123.93%, WiDeep by 162.69%, and LT-KNN by 59.3%. Similarly, in building 2, STELLAR achieves improves upon ANVIL by 88.6%, VITAL by 77.73%, SANGRIA by 165%, WiDeep by 165%, and LT-KNN by 74.9%.

4.9. CONCLUSIONS

In this chapter, we introduced STELLAR, a novel re-calibration-free framework that extends the capabilities of the ANVIL framework and demonstrates greater resilience towards heterogeneity and temporal variations in indoor localization. Through extensive experimentation, we have shown that STELLAR achieves superior localization performance across multiple heterogeneous devices and across time periods, ranging from seconds to two years. Our results indicate that STELLAR outperforms state-of-the-art localization frameworks by delivering performance improvements of 8 to 75% in terms of heterogeneity and 18 to 153% in terms of temporal variations. STELLAR also outperforms state-of-the-art localization frameworks using extended devices and timeframes by delivering performance improvements from 68 to 165%. These evaluations were conducted using real-world data and indoor environment scenarios, highlighting the effectiveness of our framework.

CHAPTER 5

SANGRIA: STACKED AUTOENCODER NEURAL NETWORKS WITH GRADIENT BOOSTING FOR INDOOR LOCALIZATION

Indoor localization is a critical task in many embedded applications, such as asset tracking, emergency response, and real-time navigation. In this chapter, we propose a novel fingerprinting-based framework for indoor localization called SANGRIA that uses stacked autoencoder neural networks with gradient boosted trees. Our approach is designed to overcome the device heterogeneity challenge that can create uncertainty in wireless signal measurements across embedded devices used for localization. We compare SANGRIA to several state-of-the-art frameworks and demonstrate 42.96% lower average localization error across diverse indoor locales and heterogeneous devices.

5.1. INTRODUCTION

Indoor localization involves the use of various technologies to determine the location of a device or person inside an enclosed space. Localization across indoor locales can serve several use cases, such as asset tracking, emergency response, and real-time indoor navigation [50]. As GPS signals are not able to penetrate buildings and other indoor environments, radio frequency signals such as Bluetooth low energy (BLE) and Wi-Fi are often utilized for localization [33]. One of the most promising methods for indoor localization involves measuring the received signal strength (RSS) of Wi-Fi signals, transmitted by Wi-Fi access points (APs), across multiple locations in an indoor floorplan. The measured RSS values at any indoor location forms a unique ‘fingerprint’ for that location that can then be used by a device to estimate location. This approach is popularly known as Wi-Fi RSS fingerprinting-based indoor localization [37].

In the fingerprinting approach, there are typically two phases: offline and online. During the offline phase, the RSS of Wi-Fi APs is captured at locations throughout the indoor space using one or more devices, such as smartphones and smartwatches. The resulting Wi-Fi RSS fingerprint data at each location is used to create a fingerprint database. A machine learning (ML) model, such as a neural network, can then be trained with the fingerprint database, such that the observed RSS fingerprint is the input, and the location is the output of the model. In the online phase, the trained ML model is deployed and used for localization. During this phase, a device measures the RSS of the Wi-Fi APs at an unknown indoor location, and the measured RSS values are sent to the ML model which in turn predicts the current location of the device.

There are several factors that can affect the accuracy of indoor localization frameworks [37]. Indoor environments are often highly cluttered, with many static and dynamic obstacles that cause unforeseeable wireless signal attenuation, as well as multipath fading and shadowing. These factors introduce noise in measured fingerprints, which can degrade the localization accuracy. Another major challenge arises due to heterogeneity in device characteristics, e.g., wireless radios that are used to transmit and receive Wi-Fi data, and software used to process the Wi-Fi signals, such as the algorithms used to filter out noise and errors [51]. Because of device heterogeneity, two different devices may end up capturing very different RSS values at the same indoor location. This phenomenon leads to reduction in indoor localization accuracy across devices. Consequently, it is crucial to devise indoor localization frameworks that can not only cope with uncertainties across indoor environments, but also factor in the impact of user device heterogeneity.

5.2. CONTRIBUTIONS

In this chapter, we present SANGRIA, a novel neural network-based Wi-Fi fingerprinting framework for indoor localization that is specifically designed to address the device heterogeneity challenge. Our approach combines the benefits of a stacked autoencoder neural network and gradient boosted trees, resulting in a highly accurate and robust approach for indoor localization. Our novel contributions are as follows:

- We propose a novel and uniquely tailored data augmentation technique for handling device heterogeneity;
- We propose a framework with gradient boosting algorithms to minimize indoor localization error;
- Through capturing fingerprints from several smartphones across diverse buildings, we establish real-world benchmarks to compare SANGRIA’s localization accuracy against several state-of-the-art localization frameworks.

5.3. RELATED WORK

Wi-Fi RSS fingerprinting is a particularly promising method for indoor localization due to the ubiquitous use of Wi-Fi capable devices by users and availability of Wi-Fi APs in indoor environments [33]. Several studies have applied ML algorithms to learn patterns in Wi-Fi RSS data and provide accurate localization predictions [37].

The work in [46] employs k-nearest-neighbors (KNN) and proposes an RSS averaging technique to improve localization accuracy but fails to show consistent results across buildings. In [52], random forests (RF) and support vector machines (SVM) are enhanced by incorporating principle component analysis (PCA) for feature extraction. The work in [17] combines gaussian

process classifiers (GPC) with an augmentation strategy to deal with noisy RSS data, but for a simpler localization problem with lower resolution. Several recent ML techniques have used gradient-boosted trees (GBT) for indoor localization [53], [55]. The GBT algorithm in [53] is applied on BLE RSS data. However, Wi-Fi RSS is more effective for localization compared to BLE RSS due to its wide availability, high accuracy, low cost, and longer range compared to BLE [54]. The works in [55] and [56] employ a simple GBT model [10] and XgBoost (extreme gradient boosting) [56] to improve localization accuracy. However, these works do not address challenges due to the variations caused by device heterogeneity.

Neural networks, e.g., deep neural networks (DNN) [42] and convolutional neural networks (CNN) [6] have also been explored to improve Wi-Fi RSS based indoor localization. The framework in [42] aims to address device heterogeneity but shows higher localization errors compared to the approaches in [46], [56]. The work in [6] demonstrates lower localization errors than [42] but does not consider device heterogeneity. The work in [45] adapts multi-head attention neural networks for localization but shows higher worst-case localization errors for certain testing devices, lacking strategic augmentation.

To mitigate device heterogeneity, augmenting training data plays a critical role in inhibiting overfitting. A few prior works have used augmentation techniques for indoor localization. In [42], generative adversarial networks (GANs) are used as an augmentation module, while [17] employs stacked autoencoders (SAEs), and [53] employs a simple neural network.

After thoroughly analyzing the prior works, we devise SANGRIA that combines a carefully tailored augmentation technique based on Stacked Autoencoder (SAE) [17] with a powerful gradient boosting algorithm [55], [56] to achieve high accuracy in a resource-efficient manner.

5.4. SANGRIA FRAMEWORK

Our proposed SANGRIA framework consists of two main components: a greedy layer-wise stacked autoencoder (custom SAE) and a categorical gradient boosting algorithm. An overview of the SANGRIA framework is shown in Figure 39.

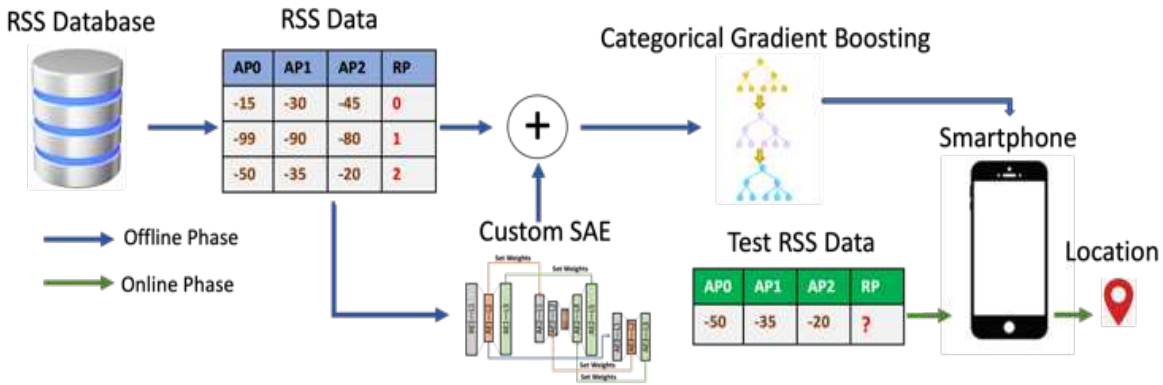


Figure 39: Implementation overview of the SANGRIA framework

In the offline phase, SANGRIA involves data collection, data pre-processing, data augmentation, and ML model training and deployment. The data collection and pre-processing phases are used to generate an RSS fingerprint database. A custom stacked autoencoder (SAE) model is tailored to augment fingerprints from the database such that an ML model can produce accurate predictions despite device heterogeneity. The collected data prior to augmentation and the augmented data are concatenated to diversify the RSS input data. The concatenated data is then used to train the categorical gradient boosting ML model, to learn the relationship between RSS measurements and indoor locations. Finally, this trained model is deployed on the smartphone. The offline phase is typically performed once, and the resulting model is used for multiple localization estimates in the online phase. The accuracy of the location estimates is determined by computing error using the Euclidian distance:

$$Distance = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2 + (Z_2 - Z_1)^2} \quad (16)$$

where (X_1, Y_1, Z_1) and $X_2, Y_2, Z_2)$ are the values of the ground truth and predicted location coordinates, respectively.

5.4.1. DATA COLLECTION, PREPROCESSING, AND DATABASE CREATION

For any indoor environment of interest, we collect Wi-Fi RSS values in dBm (decibel-milliwatts). Each RSS value ranges between -100 dBm to 0 dBm, where -100 indicates no signal and 0 indicates a full (strongest) signal. We collect RSS values from multiple (six in our experimental analysis) smartphones from different manufacturers, with different Wi-Fi chipsets. The collected data is pre-processed by normalizing and splitting it into train and test datasets for subsequent use with the ML algorithm. We normalize all RSS values between the range of 0 (-100 dBm) to 1 (0 dBm). The resulting normalized vector of RSS values for each location reference point (RP) represents a fingerprint and is stored in the RSS database. To evaluate the ML model, we split the collected RSS data into training and testing datasets.

5.4.2. GREEDY LAYER-WISE STACKED AUTOENCODER

To augment the collected data before training, we design a custom stacked autoencoder (SAE) neural network model. Our SAE model is composed of multiple autoencoders (AE), as shown in Figure 40, which are trained one layer at a time using a greedy layer-wise training approach.

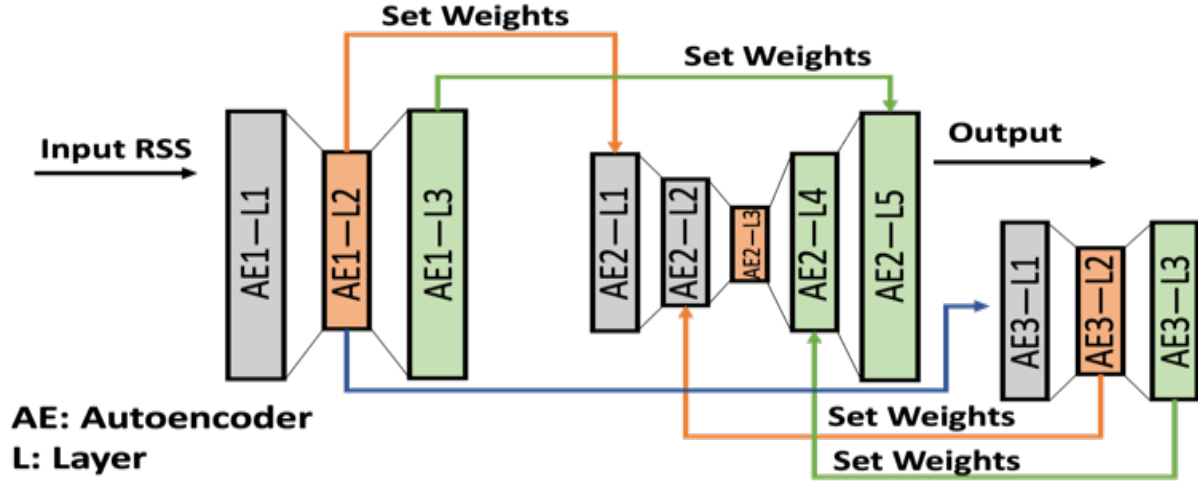


Figure 40: Greedy layer-wise stacked autoencoder (SAE) architecture with Encoder (gray), Decoder (green), and bottleneck (orange) layers.

An AE neural network consists of three main parts: an Encoder, a Decoder, and a Bottleneck layer. Each of these parts is composed of one or more fully connected layers with varying number of neurons. The encoder layers are used to compress the input data (dimensionality reduction) and the decoder layers are used to reconstruct a noisy representation of the input data, back to its original dimension. The bottleneck layer is a representation of the input data at its lowest dimension, representing important features in the data.

The proposed SAE consists of three AEs stacked together with each having its own fully connected layers, as shown in Figure 40. AE1 and AE3 are trained layer-wise, where the output of the second layer (L2) of AE1 is fed as input to AE3. AE2 receives the trained weights from AE1 and AE3 and is positioned as shown in Figure 40. The SAE is trained in a greedy layer-wise manner to minimize reconstruction error between the input and output data. The advantage of this approach is that it allows the network to learn a good representation of the input data at each layer before moving on to the next one, rather than trying to optimize all the layers together. This can make the training process more stable and can lead to better performance.

SAE is trained on the RSS database to generate synthetic RSS measurements, simulating variations across devices that are not overly biased towards the device used for training, resulting in better generalization and more accurate predictions.

5.4.3. CATEGORICAL GRADIENT BOOSTING

Gradient-boosted trees (GBTs) are a powerful type of ML model that involve training a series of decision trees and combining their predictions to create a stronger overall model. Instead of predicting the target variable, each decision tree in a gradient boosting model is trained to predict the residual error of the previous tree, which helps minimize the overall error during training. There are several types of GBT models, including CatBoost, XgBoost, and LightGBM. We chose to use CatBoost for our problem because it is better suited for dealing with categorical variables and does not require additional pre-processing steps, making it a more efficient choice. Additionally, its symmetric trees can provide more accurate predictions in cases with a large number of categorical features, such as the number of unique visible Wi-Fi APs in our indoor localization problem. To train our model, we iteratively update each decision tree to the residual error of the previous iteration, as per the mathematical representation in the equations below:

$$F(x) = T\left(\sum W_i * H_i(x)\right) \quad (17)$$

$$F(x) = F(x - 1) + H(x) \quad (18)$$

where $H_i(x)$ is the prediction made by the i^{th} decision tree, W_i are the weights assigned to the i^{th} tree, and T is monotonic transformation applied to the combined predictions. The monotonic

transformation is key as it retains the order of the RSS fingerprint for each tree (changes in the fingerprint order can lead to higher localization errors). The weights W_i are representative of the learned features from the RSS fingerprint. $H(x)$ is the prediction made by the decision tree at iteration x . $F(x - 1)$ is the prediction made by the model after $x-1$ iterations. $F(x)$ is the final prediction (location of the RSS fingerprint) made by the model after x iterations.

5.5. EXPERIMENTS

We collected fingerprint data from five buildings with different salient features on the Colorado State University campus. The buildings have path lengths varying between 60 to 80 meters and contain 80 to 180 Wi-Fi APs. RP locations for each fingerprint are one meter apart. Building 1 has wooden furniture and concrete walls, Building 2 has heavy metallic laboratory equipment, Building 3 and 4 have a mix of wood, concrete, and smaller metallic equipment and computers, and Building 5 has wide open spaces.

To account for device heterogeneity, we selected six smartphones: BLU Vivo 8 (BLU), HTC U11 (HTC), Samsung Galaxy S7 (S7), LG V20 (LG), Motorola Z2 (MOTO), and OnePlus 3 (OP3). During the training phase, we collected data from a single smartphone per building floorplan. We ensured that each smartphone captured between 61 to 78 unique RPs per floorplan, with 78 to 339 visible APs. We also took five samples per RP for training and one sample per RP for testing in total, we collected 1606955 data points for our training dataset, which includes all floorplans. To further improve the size and quality of our training dataset, we utilized the custom SAE for data augmentation. The SAE effectively doubles the size of our training dataset, resulting in a total of 3213910 data points for training. We collected the RSS fingerprints in a real-world scenario and considered building floorplans with multiple salient features, including external

obstacles that can add noise to the collected RSS data. Moreover, we collected training and testing data during normal working hours of the day to incorporate environmental factors such as human interference, dynamic obstacles, signal attenuation, fading, shadowing, and so on.

A detailed description of the number of trainable parameters in the proposed SAE within SANGRIA is shown in Table 6. We enhanced the baseline CatBoost algorithm from the library in [57]. To fine-tune CatBoost within SANGRIA, we set number of iterations to 50, with a total depth in trees to 7, learning rate of 0.1, and L2 leaf regularization to 5 to prevent overfitting. In the next section we present experimental results of SANGRIA’s performance comparison against the state-of-the-art.

Table 6: Trainable parameters for each stack in the SAE in SANGRIA

AE1-L1	AE1-L2	AE1-L3	AE2-L1	AE2-L2	AE2-L3
14878	10414	14964	10414	5103	3570
AE2-L4	AE2-L5	AE3-L1	AE3-L2	AE3-L3	Total
7388	14964	7289	5103	7388	101475

5.5.1. EXPERIMENTAL RESULTS

To evaluate the performance of SANGRIA, we select seven state-of-the-art indoor localization frameworks for comparison: KNN [46], RF [52], SVM [52], GPC [17], and DNN [42] that target localization with device heterogeneity resilience; and XgBoost [56] and CNN [6] which aim to aggressively minimize indoor localization error but do not address device heterogeneity.

We collect localization errors for each train-test combination across all paths. The results are plotted in Figure 41 by averaging the localization errors across all floorplans. From Figure 41, we can clearly observe low errors for SANGRIA compared to other frameworks. SANGRIA shows

28% to 66% improvements in mean, and 42% to 74% maximum error reduction, compared to the other frameworks. This is an indication of SANGRIA’s excellent generalizability across all devices.

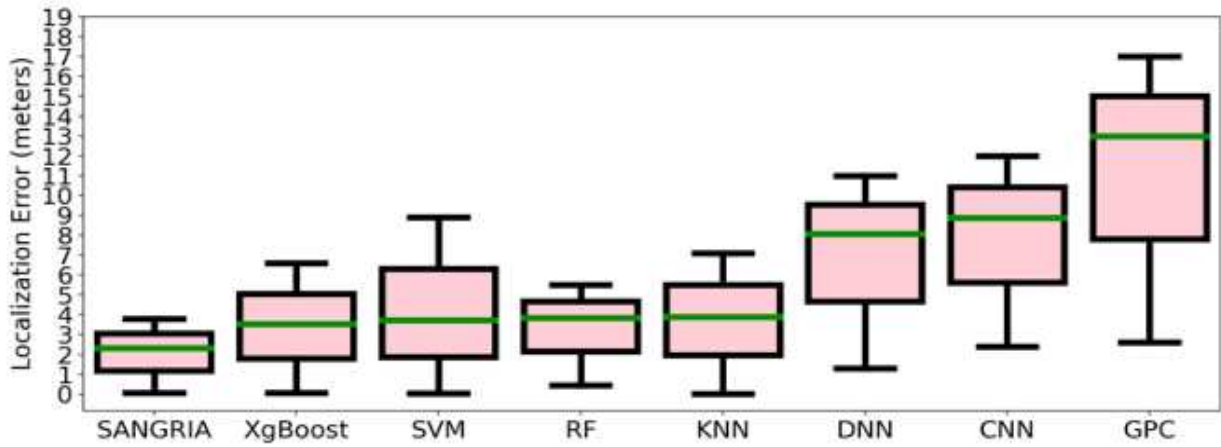


Figure 41: Min (lower whisker), Mean (green bar), and Max (upper whisker) error across all comparison frameworks for indoor localization.

To further analyze the individual device-wise performance of each framework, we construct a heatmap for the top four best performing frameworks, as shown in Figure 42. The error (in meters) for each cell is the average for that configuration (train-test combination) across all five building floorplans. From Figure 42, we see that SANGRIA improves the worst-case localization errors compared to the other frameworks. For example, in the LG-MOTO and LG-OP3 configurations, we observe higher errors in SVM and XgBoost, but these worst- case errors are greatly reduced for SANGRIA. Similarly, in the BLU-OP3 configuration, we observe high errors in RF and these errors are reduced for the rest of the frameworks with SANGRIA reporting the lowest error.

We have open sourced our dataset [59] to aid reproducibility. To further evaluate the effectiveness of SANGRIA, we trained the proposed ML models using another publicly available dataset: the UJI dataset [58] for indoor localization. The dataset contains Wi-Fi RSS measurements

from 520 APs distributed across a three-story building. Each of the 520 Wi-Fi APs was equipped with one receiver and four antennas, and data was collected using 20 Wi-Fi cards. Figure 43 shows the mean localization errors of each model trained and tested on the UJI dataset. Notably, SANGRIA has the lowest localization errors, surpassing its closest competitor XgBoost by 11.4%.

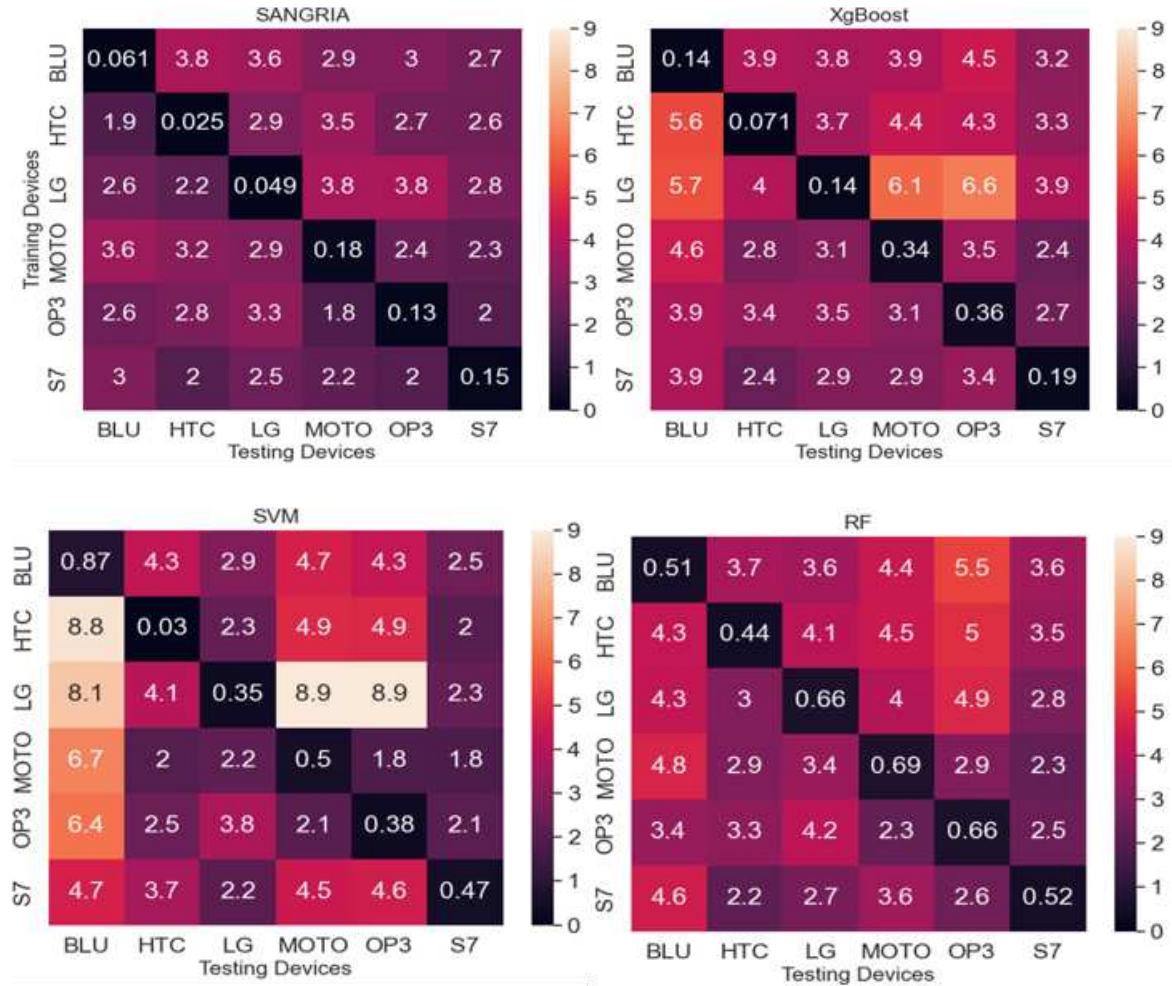


Figure 42: Experimental results for the top four best performing frameworks with average errors (meters) across all floorplans.

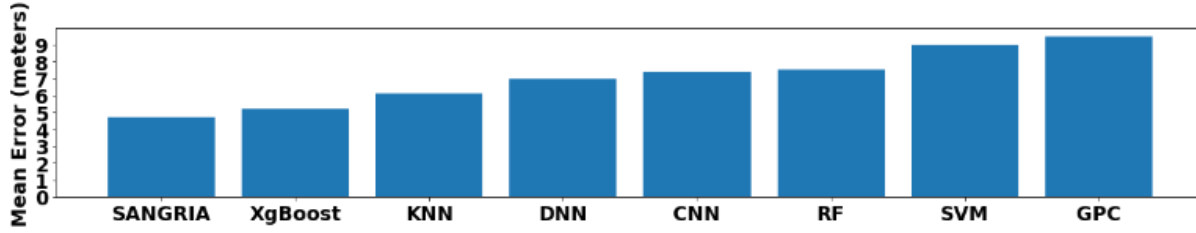


Figure 43: Mean localization errors across all frameworks using UJI dataset

SANGRIA’s superior localization capabilities are to a large extent due to the uniquely tailored greedy-layer wise SAE. The other contributing factor is the excellent feature extraction ability of the CatBoost algorithm. CatBoost has several advantages over other algorithms due to its ability to handle categorical features, such as number of visible APs and MAC (Media Access Control address) ID of each Wi-Fi AP. In contrast, other algorithms such as DNN, CNN, SVM, RF, KNN, and GPC require the categorical features to be transformed into numerical form. CatBoost also can better reduce overfitting, which is a common problem in GBT algorithms. It does this by randomly selecting a subset of features to use at each iteration of the boosting process, making it a better choice than XgBoost.

5.5.2. EFFECT OF THE STACKED AUTOENCODER

In Figure 44, we observed a significant drop in mean localization errors for all frameworks when using SAE: 6.82% lower mean error for CatBoost, 17.64% lower mean error for XgBoost, 20.12% lower mean error for SVM, 8.16% lower mean error for RF, 2.64% lower mean error for KNN, 37.94% lower mean error for DNN, 47.34% lower mean error for CNN, and 87.59% lower mean error for GPC.

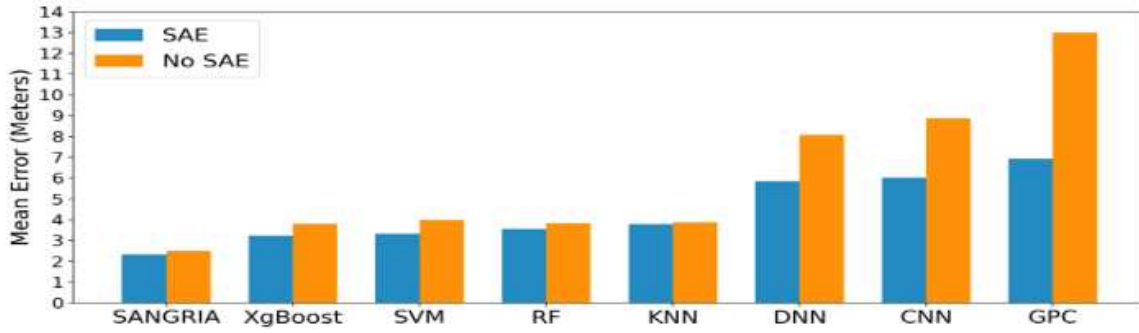


Figure 44: Bar plot comparing average errors of all frameworks with SAE (Blue) and without SAE (Orange)

5.5.3. MODEL INFERENCE TIME LATENCY

Low latency in the online phase is important to achieve practical indoor localization framework deployment. Table 7 shows the measured latencies in milliseconds for each framework on smartphones. SANGRIA has the fastest inference time with an average of ~13.3ms, followed by KNN with ~21.6ms, and XgBoost with ~20ms. GPC has the highest latency due to its probabilistic computation.

Table 7: Avg. inference time (milliseconds) across devices and frameworks

Devices	SANGRIA	XgBoost	SVM	RF	KNN	DNN	CNN	GPC
BLU	20	30	400	90	30	630	680	1960
HTC	10	20	110	80	20	540	630	1830
LG	20	30	200	80	10	550	630	1860
MOTO	10	10	190	70	20	540	610	1820
OP3	10	20	280	80	30	540	620	1810
S7	10	10	180	80	20	510	610	1740
Avg.	13.3	20	226.6	80	21.6	551.6	630	1836.6

5.6. CONCLUSION

In this chapter, we present SANGRIA, a novel neural network-based solution that is resilient towards device heterogeneity during indoor localization. Experimental results show a 42.96% reduction in average localization error and an average latency of 13.3 milliseconds, making it a reliable and practical approach for device-invariant indoor localization.

CHAPTER 6

FEDHIL: HETEROGENEITY RESILIENT FEDERATED LEARNING FOR ROBUST INDOOR LOCALIZATION WITH MOBILE DEVICES

Indoor localization plays a vital role in applications such as emergency response, warehouse management, and augmented reality experiences. By deploying machine learning (ML) based indoor localization frameworks on their mobile devices, users can localize themselves in a variety of indoor and subterranean environments. However, achieving accurate indoor localization can be challenging due to heterogeneity in the hardware and software stacks of mobile devices, which can result in inconsistent and inaccurate location estimates. Traditional ML models also heavily rely on initial training data, making them vulnerable to degradation in performance with dynamic changes across indoor environments. To address the challenges due to device heterogeneity and lack of adaptivity, we propose a novel embedded ML framework called FedHIL. Our framework combines indoor localization and federated learning (FL) to improve indoor localization accuracy in device-heterogeneous environments while also preserving user data privacy. FedHIL integrates a domain-specific selective weight adjustment approach to preserve the ML model’s performance for indoor localization during FL, even in the presence of extremely noisy data. Experimental evaluations in diverse real-world indoor environments and with heterogeneous mobile devices show that FedHIL outperforms state-of-the-art FL and non-FL indoor localization frameworks. FedHIL is able to achieve $1.62\times$ better localization accuracy on average than the best performing FL-based indoor localization framework from prior work.

6.1. INTRODUCTION

Indoor localization is the process of determining the location of a device or user within an indoor environment, such as a building or a warehouse [4]. This technology is becoming increasingly important in critical applications such as emergency response, security, and healthcare, where knowing the precise location of individuals or assets can be critical for saving lives and preventing harm [137]. Many efforts from industry and academia are attempting to build indoor localization solutions because of its promising future [34]. One foundational technology for indoor localization is Wi-Fi, which is ubiquitously deployed across indoor locales and can thus be cost-effectively used as part of indoor localization solutions [3]. There are two main types of Wi-Fi-based indoor localization systems: propagation model-based and fingerprinting-based [138], [141].

Propagation model-based methods estimate the distance between the transmitter (e.g., Wi-Fi access point (AP)) and receiver (e.g., mobile device carried by user) using propagation models such as the free space model. Then, by computing the relative locations of the receiver to specific transmitters, these methods estimate the user's coordinates in the indoor environment. Propagation model-based methods have advantages but also many limitations. These methods require prior knowledge of the AP locations, which may make them unsuitable for environments where AP locations are unknown or deliberately obscured due to security concerns [142]. Additionally, signal propagation in the indoor environment is dynamic and influenced by unpredictable factors such as obstacles and people, which can impact the accuracy of propagation model-based methods. Other factors, such as signal attenuation, multi-path fading, and shadowing, can also distort signals and result in inaccurate location estimates [143]. These limitations motivate the need for more robust localization methods.

Fingerprinting-based methods extract representative and distinguishable characteristics from indoor environments, called fingerprints [144]. In these methods, any radio frequency (RF) signal, such as Wi-Fi, Bluetooth, and Zigbee, can be used to estimate the user's location in indoor environments [145]. However, Wi-Fi is one of the most popular and widely used RF signals for indoor localization due to its availability in almost all indoor locales, without the need for deploying expensive custom APs [4], [137], [34]. Compared to other approaches, Wi-Fi fingerprinting can provide more accurate location estimates as it has a higher sampling rate and can enable capturing a larger amount of signal data. Additionally, Wi-Fi signals can penetrate through walls and other obstacles, providing more comprehensive coverage of indoor environments [146].

A Wi-Fi fingerprinting system typically consists of an offline and an online phase [3], [4], [34], [137]. In the offline phase, Wi-Fi signal characteristics (e.g., received signal strength or RSS) at predefined reference points (RPs; representing specific locations within the indoor environment) are collected to build a fingerprint database. In the online phase, the system extracts a fingerprint from the observed Wi-Fi signals in the same way and matches the observed fingerprint to those saved in the database. The user's location is then estimated by determining the RP whose fingerprint is most similar to the observed fingerprint captured by the user's mobile device. While Wi-Fi fingerprinting has many advantages, there are also several limitations [142], [143]. One major limitation is device heterogeneity, which can result in variations in signal strength and accuracy [143], [146]. Different mobile devices can have varying Wi-Fi chips, antennas, and software stacks that can impact the captured Wi-Fi signal characteristics. This can lead to inconsistencies in the Wi-Fi fingerprints collected from different devices, making it difficult to accurately estimate the user's location. Additionally, changes in the Wi-Fi environment, such as

interference from movement of people or changes in the configuration of the Wi-Fi APs can also impact the accuracy of the location estimates [146].

Machine learning (ML) models can be trained to learn features from Wi-Fi data (e.g., RSS) and use them to provide more accurate location predictions and address some of the limitations mentioned above [36]. One approach is to use supervised learning, where the system is trained using a dataset of known locations (RPs) and their corresponding Wi-Fi RSS fingerprints. In this approach, the ML model learns to map the input (Wi-Fi RSS fingerprint) to the output (known RP). The ML model structure can vary, e.g., they can be fully connected deep neural networks or convolutional neural networks [147]. Another approach is to use unsupervised learning, where the system is not provided with labeled data but instead tries to identify patterns and structure in the data [148]. In this approach, the ML model can be used to cluster similar Wi-Fi RSS fingerprints together, which can help to identify regions of similar signal strength in the indoor environment [149]. This information can be used to provide location estimates based on the nearest cluster or the centroid of the cluster.

Another approach to training ML models for indoor localization is in a distributed manner by using federated learning (FL). This approach allows multiple devices to collaboratively train a model without sharing their data, which is particularly important in scenarios where data privacy is a concern, such as in healthcare or industrial tracking [150]. In FL, the devices do not share their location data directly with a central server, but instead share the updated model parameters from a local model (LM) with a server, which aggregates the received parameters from each device to update a global model (GM) maintained at the server, while keeping the data on each device private [151]. FL can be used in scenarios where devices are heterogeneous, i.e., with different capabilities or hardware [152]. The FL approach allows for each device to contribute to the model training,

regardless of its capabilities, providing a more comprehensive and diverse dataset. But the FL technique also has a major limitation: the performance of FL depends on the quality and diversity of the data contributed by each device [122], [123], [124]. If the data provided by different devices is noisy, it can lead to a degradation of the model performance, as the model may struggle to generalize across the data. This is because data from different devices may contain noise or outliers that fall outside the true distribution of the data. In such cases, effective feature selection of the trained model parameters before aggregating them becomes extremely important.

6.2. CONTRIBUTIONS

In this chapter, we propose a novel embedded ML framework called FedHIL, that utilizes a neural network-based FL algorithm tailored for the indoor localization domain, to address the multiple challenges of device heterogeneity, model degradation, and privacy concerns. Our key contributions are as follows:

- We propose a novel and strategically tailored data augmentation strategy using unsupervised learning to address the variations in the Wi-Fi RSS fingerprints caused by device heterogeneity.
- We propose a combination of a lightweight neural network (supervised learning) within an FL setting that continually updates the underlying supervised model to maintain accuracy within the device heterogeneity-resilient and privacy-preserving framework.
- We propose a unique and strategic federated aggregation technique to contribute to the robustness of the proposed framework for indoor localization, thereby preventing degradation of the model's performance, caused by noisy or outlying data.
- We deploy the proposed FedHIL framework on multiple heterogeneous mobile devices and conduct evaluations across real-world building environments, and comprehensively compare

its performance against several state-of-the-art FL- and non-FL based indoor localization frameworks.

6.3. RELATED WORK

In recent years, ML has become increasingly popular in Wi-Fi RSS fingerprinting based indoor localization [118]. Supervised learning methods such as K-nearest neighbor (KNN) [153], support vector machines (SVM) [154], random forest (RF) [155], and neural networks including deep neural networks (DNN) [78] and convolutional neural network (CNN) [79] have been widely used to improve localization accuracy. In particular, neural network-based approaches have shown promising results due to their ability to model complex non-linear relationships in the data [78], [79]. However, these supervised learning methods do not address the issues faced due to device heterogeneity. These methods also require a large amount of training data, which is problematic in real-world settings, where collecting large amounts of labeled fingerprint data may be too time consuming and costly, and thus impractical.

To overcome the limitations of supervised learning-based approaches, some prior works employ unsupervised learning-based approaches such as principal component analysis (PCA) [156], k-means clustering [157], and non-negative matrix factorization (NMF) [158]. These approaches aim to discover the underlying structure in the Wi-Fi RSS fingerprint data without the need for labeled data (RP of the corresponding fingerprints). Such methods can be useful in cases where obtaining labeled training data is difficult or costly. The works in [156], [157], [158] utilize various unsupervised learning methods to extract the most important features from the data and represent it in a lower-dimensional space. This can help to improve the performance of the ML model by reducing the complexity of the data and removing irrelevant or redundant features. While

unsupervised methods can be useful for discovering underlying structure in the data, they are rarely as accurate as supervised methods in predicting location. Therefore, recent works propose a combination of supervised and unsupervised learning to create a more robust model to address device heterogeneity.

The hybrid approaches that combine supervised and unsupervised learning-based models take advantage of the strengths of the two unique learning approaches to improve the accuracy and robustness of indoor localization. The unsupervised learning methods are typically used to augment the training data, thereby diversifying the training data [159]. Augmenting the data can be beneficial because it can result in physically capturing lesser data for training while maintaining the diversity in the data. Additionally, augmenting the data can help reduce the effects of heterogeneity by generating more training examples with variations in the RSS. This improves the robustness of the model and its ability to handle RSS variations caused by device heterogeneity. The works in [82], [160], [161], [162] make use of hybrid approaches. They concatenate the output from the unsupervised learning models and send it to a supervised learning model to yield stable localization results. This approach allows the supervised learning model to learn from the augmented data and extract the most relevant features from the data.

In recent years, many other frameworks have been proposed to address the challenge of device heterogeneity during indoor localization [45], [46], [80], [136], [163]. These frameworks employ a variety of techniques to address device heterogeneity, including multi-headed attention neural networks with a lightweight augmentation strategy [45], [163], a combination of KNN and DNN [36], vision transformer neural networks with specialized augmentation strategies [80], [136], and stacked autoencoders with a Gaussian process classifier (GPC) [17]. While these frameworks have shown promising results, they heavily rely on the initial training phase of the model, and any noisy

or outlier data during the online phase can negatively impact the accuracy of the localization prediction.

A few recent works explore the use of federated learning (FL) to achieve collaborative learning for indoor localization [164], [165], [166]. As discussed earlier, FL allows multiple participating client devices to train an underlying GM at a server by sharing parameters from LMs on these devices that are trained with locally collected data. This approach preserves privacy during distributed learning as devices do not need to share their local (fingerprint) data with the server. The work in [164] proposes a centralized indoor localization method using pseudo-labels (CRNP). The authors use a stacked autoencoder for data augmentation and a DNN-based classifier with FL. During the online phase, the pseudo-labels generated from the DNN are used to retrain the model, and the retrained parameters are aggregated using the federated averaging (FedAvg) [167] method, which is a popular method for aggregation in FL. FedAeDNN [165] uses an autoencoder and DNN with FedAvg, while [166] proposes FedLoc, which uses a multi-layer perceptron and a federated stochastic gradient descent (FedSGD) [168] aggregation strategy. FedSGD is similar to FedAvg but uses a stochastic gradient descent (SGD) approach for model aggregation.

The ability to learn from a greater amount of data (from multiple clients while preserving their data privacy) has allowed these FL-based approaches to achieve promising results. However, these methods are sensitive to noisy or outlier data, which can skew the model's performance and lead to poor generalization and overfitting. To overcome these issues, in this work we propose a novel FL-based framework called FedHIL that builds on the merits of the FL-based methods discussed earlier. FedHIL utilizes a novel and uniquely tailored unsupervised learning-based ML model for efficient data augmentation, and the augmented data is further trained with an underlying lightweight supervised model that is also robust to device heterogeneity. This proposed

combination is used in a distributed FL environment, where multiple client devices are used to update the model with their new and heterogeneous data. As FL can cause the accuracy of the underlying ML model to degrade, we further propose a new domain-specific selective weight adjustment approach to preserve the ML model's performance for indoor localization, even in the presence of extremely noisy data.

6.4. HETEROGENEITY ANALYSIS OF WI-FI RSS FINGERPRINTS

Wi-Fi RSS is a measurement of the power level of the wireless signal received from a Wi-Fi AP by a Wi-Fi receiver, such as those found in all smartphones, Tablets, and laptops today. The RSS value is expressed in units of decibels (dB) relative to a milliwatt and represents the strength of the Wi-Fi signal received by the receiver. These values typically range from -100dB to 0dB [63]. A value of -100dB indicates that no signal is received, while a value of 0dB represents the strongest signal strength that can be received. However, it is rare to capture an RSS value of 0dB in real-world environments, as the signal strength is typically affected by factors such as signal attenuation, reflection, and multipath propagation. In practice, the RSS values captured during Wi-Fi fingerprinting are often in the range of -80dB to -30dB, with values below -70dB generally considered weak and values above -50dB generally considered strong [63]. The exact threshold values for strong and weak RSS signals may vary depending on the specific Wi-Fi chipset and the environment in which it is used.

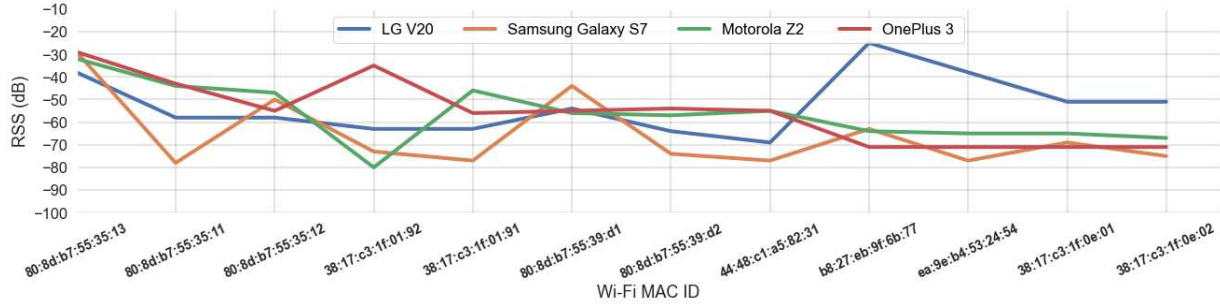


Figure 45: RSS fingerprints of visible Wi-Fi APs collected using four different smartphones at the same indoor location.

The observed Wi-Fi RSS values at any indoor location are impacted by many factors, such as the distance between the receiver and the AP, the type of obstacles and interference in the signal path, the signal propagation characteristics of the wireless channel, and most importantly, device heterogeneity. Device heterogeneity is a major problem in indoor localization because it significantly affects the accuracy and reliability of localization results. This is because indoor localization systems rely on Wi-Fi RSS fingerprints to estimate the location of the target device, and the RSS values captured by different devices can vary significantly due to the differences in their hardware, software, and antenna configurations [146]. Devices with the same Wi-Fi chipset can exhibit different Wi-Fi RSS readings due to heterogeneity in their firmware stacks.

Variations in firmware configurations, calibration settings, and signal processing algorithms can lead to discrepancies in the RSS values reported by different devices, even if they share the same underlying Wi-Fi chipset. Since the RSS values captured by different devices at the same location can vary, the data used to train ML models that are part of indoor localization frameworks may not accurately represent the precise RSS values incident at a location. This can result in poor generalization of the model, leading to inaccurate localization results. To better understand device heterogeneity-based variations, we show results from a study we conducted that captured Wi-Fi

RSS values at a specific indoor location in a real-world building setting with four different smartphones: LG V20, Samsung Galaxy S7, Motorola Z2, and OnePlus 3, as shown in Figure 45. The x-axis represents the MAC IDs (Media Access Control address) of the Wi-Fi APs, and the y-axis shows the RSS values captured by each smartphone. Ideally, the RSS value from each AP should be the same across the smartphones. From Figure 45, we observe that the values can differ significantly. For instance, for the AP with the MAC address '80:8d:b7:55:35:11', the LG V20 recorded an RSS value of -58 dB, while the Samsung Galaxy S7 recorded -78 dB. Similarly, for the AP with the MAC address '38:17:c3:1f:01:92', the Motorola Z2 and Samsung Galaxy S7 recorded -80 dB and -73 dB respectively, while LG V20 and OnePlus 3 recorded -63 dB and -35 dB respectively. We comprehensively analyzed such RSS variations across different buildings and a larger pool of heterogeneous smartphones. We utilized this information to both design and validate our FedHIL framework. Our data augmentation and FL strategies have been specifically designed to improve the accuracy and robustness of the indoor localization system by mitigating the impact of RSS variations.

6.5. BACKGROUND ON FEDERATED LEARNING (FL) FOR INDOOR LOCALIZATION

FL is a powerful technique that enables training of ML models across multiple decentralized client devices. In an FL system, training data is kept on the local devices (decentralized client devices), and the ML model is trained on each device. The updated model parameters are then transmitted and aggregated to create a global model that reflects the collective knowledge of all the devices. This approach provides many benefits, including enhanced data privacy, improved scalability, and long-term reliability of the model. In the context of indoor localization with Wi-Fi

RSS data, FL can be used to train models that accurately predict the location of a device based on its local Wi-Fi RSS readings. Figure 46 shows an overview of how FL can be utilized for indoor localization. In this scenario, a centralized server is used to maintain a global model (GM), which is pretrained with a subset of the Wi-Fi RSS data. The GM represents the collective knowledge of all the local client devices participating in the FL process. Initially, this GM is first sent to client devices (e.g., via a localization app) which use it to make predictions based on their local Wi-Fi RSS readings. After receiving the GM from the centralized server, a client device can retrain the GM with its own local RSS data to create a local model (LM). The local training process involves retraining the received GM from the server using the client device's own RSS fingerprint dataset.

As the client device's local RSS data is used to update the weights (model parameters) of the LM, it helps to improve the accuracy of the GM by accounting for local variations in the Wi-Fi RSS data. To collect the local Wi-Fi RSS readings, a client device at a specific indoor location captures Wi-Fi signals from nearby APs and measures their strength. The RSS values are combined into an RSS fingerprint, which is then sent to the LM to estimate the device (and user's) location. The client device can display the location on the user's device on a map of the building floorplan and this can be used for a variety of applications such as indoor navigation, location-based advertising, and asset tracking. The trained parameters from the client device's LM are periodically sent back to the centralized server, where they are aggregated with the original GM and parameters received from other devices, to update the GM, which is then sent to all client devices (Figure 46). This process of updating the GM with the contributions from multiple client devices helps to achieve better accuracy and scalability than traditional ML approaches.

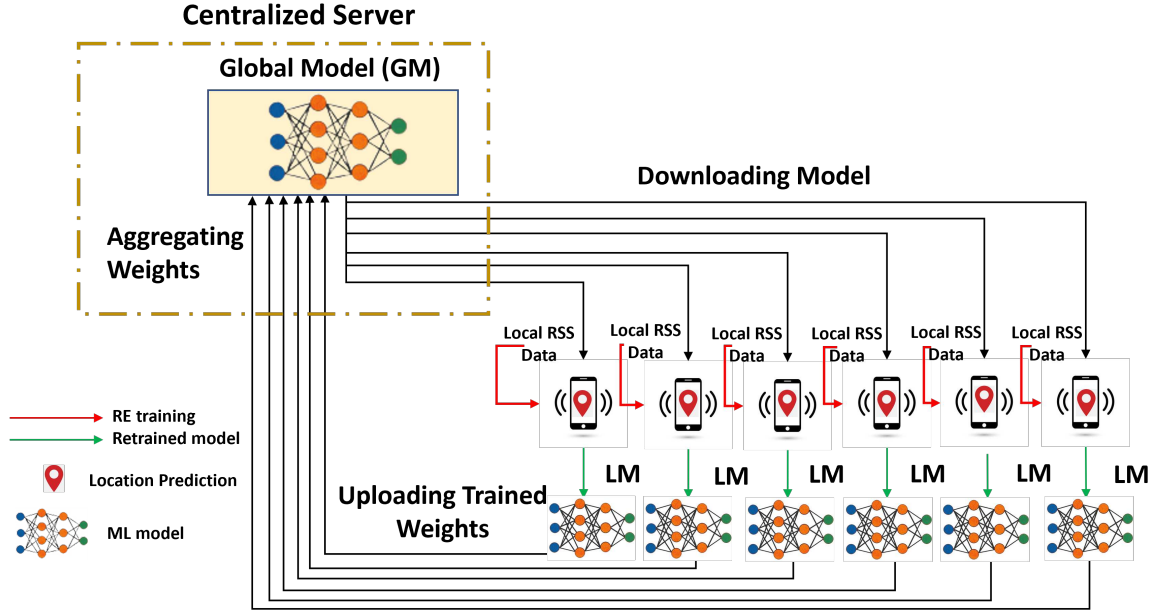


Figure 46: Federated learning (FL) overview using multiple client devices for indoor localization.

While there are several popular aggregation techniques used in FL, state-of-the-art approaches that use FL for indoor localization ([164], [165], [166], discussed earlier in Section 6.4) employ FedAvg [167] and FedSGD [168]. These aggregation techniques offer a simple and efficient way to combine models from different clients. We describe these aggregation approaches next. In Section 6.6, we describe our proposed FL-based localization framework.

6.5.1. FEDERATED AVERAGING (FEDAVG)

The FedAvg algorithm, first introduced by [167] in 2016, aims to train a GM using data available on multiple devices without transferring the data to a central server. The algorithm can be broken down into six steps: 1) initialization of a GM on a central server, 2) selection of a subset of clients to participate in each round of training, 3) sending of the GM to the selected client devices, 4) local training of the model on each device using an SGD algorithm, 5) sending of the

updated models back to the central server, and 6) aggregation of the models using a weighted average. The aggregation strategy used in the FedAvg technique is shown in the equations below.

$$W_{client} = W_T - \eta * \nabla lf(W_T) \quad (19)$$

$$W_{global} = \sum_{i=1}^N \frac{K_i + W_{client}(i)}{K} \quad (20)$$

where W_{client} are the weights generated from the LM using SGD, W_T are the weights from the GM at the beginning of the round, η is the learning rate, and $\nabla lf(W_T)$ is the gradient of the loss function with respect to the GM weights. The W_{client} obtained from Equation 19 is used to update the GM using the weighted average method, as shown in Equation 20. W_{global} are the updated GM weights, N is the number of client devices, K_i is the number of local data samples of the i^{th} client, and K is the total number of local data samples in all clients. Equation 20 calculates the weighted average of the LMs based on the number of local data samples available at each client. Clients with more data contribute more to the GM, while clients with less data have a smaller impact on the final model.

FedAvg is a popular technique for training ML models in distributed settings. However, it suffers from limitations such as overfitting and low accuracy over time in the presence of high noise in environments. For applications such as indoor localization using Wi-Fi RSS data, the data can be preprocessed and cleaned on each client device, and local ML models can be trained using an SGD algorithm. The resulting models can be aggregated using the FedAvg algorithm to predict the location within an indoor environment without the need for centralized data storage or

processing. Overall, FedAvg is a powerful technique that can be applied to a wide range of applications, including those involving distributed data.

6.5.2. FEDERATED STOCHASTIC GRADIENT DESCENT (FEDSGD)

FedSGD [168] was introduced as another distributed ML technique, similar to FedAvg. The main difference between FedSGD and FedAvg is the aggregation strategy, where FedAvg uses a weighted average of the LMs, while FedSGD uses a simple averaging approach. Like FedAvg, FedSGD also follows the same steps, except for the aggregation step. In FedSGD, gradients are calculated locally on individual devices using available data, which are sent to a central server for aggregation and to update the GM. Gradients represent the direction and magnitude of change required to minimize the loss function, which is used to update the weights in each iteration of the training process. The equations below capture the aggregation strategy used in FedSGD.

$$G_{client} = \nabla l f(W_T) \quad (21)$$

$$G_{avg} = \frac{1}{N} \sum_{i=1}^N G_{client(i)} \quad (22)$$

$$W_{global} = W_T - \eta * G_{avg} \quad (23)$$

where G_{client} are the gradients of the client models' loss functions with respect to the GM's weights, which is represented as W_T . These gradients are sent to the server for aggregation. The G_{avg} are the averaged gradients of all the client devices. N is the number of client devices and $G_{client(i)}$ are the gradients computed using Equation 21 per client. W_{global} are the new GM weights obtained by subtracting the product of the learning rate (η) and the average of the client

gradients (G_{avg}) obtained from Equation 22. In this way, FedSGD technique uses the client gradients instead of the client weights (as done in FedAvg) to compute the average.

Much like FedAvg, FedSGD is beneficial in training ML models on decentralized and heterogeneous data sources, while maintaining privacy. In indoor localization based on Wi-Fi RSS data, FedSGD can be suitable for scenarios with a small number of clients and less imbalanced (or noisy) data. It can also be useful when communication bandwidth is limited, as its communication cost is lower than FedAvg.

6.6. FEDHIL FRAMEWORK

In this section, we describe our proposed FedHIL framework for FL-based indoor localization. The framework follows the general principles of FL, as discussed earlier. However, it also includes modifications that have been tailored to address some of the key challenges associated with device heterogeneity and long-term robustness. The first modification is in the GM architecture, which allows the framework to aggregate data from different sources and train a model that can generalize well across different devices. The second modification is in the LM architecture, which enables each device to train its model on local data while ensuring that the data remains private and secure. Finally, the third modification is in the federated aggregation approach, which ensures that the GM is updated with new information from different LMs via communication between the LM and GM. By integrating these modifications into the FL framework, FedHIL aims to overcome some of the limitations of traditional FL methods for the indoor localization problem. In the following sections, we describe each of these mechanisms, providing a comprehensive overview of the FedHIL framework and its capabilities.

6.6.1. GLOBAL MODEL (GM) ARCHITECTURE

The GM architecture plays a critical role in achieving high accuracy and scalability. The GM architecture consists of two phases, namely, the offline training and online training phase, as shown in Figure 47.

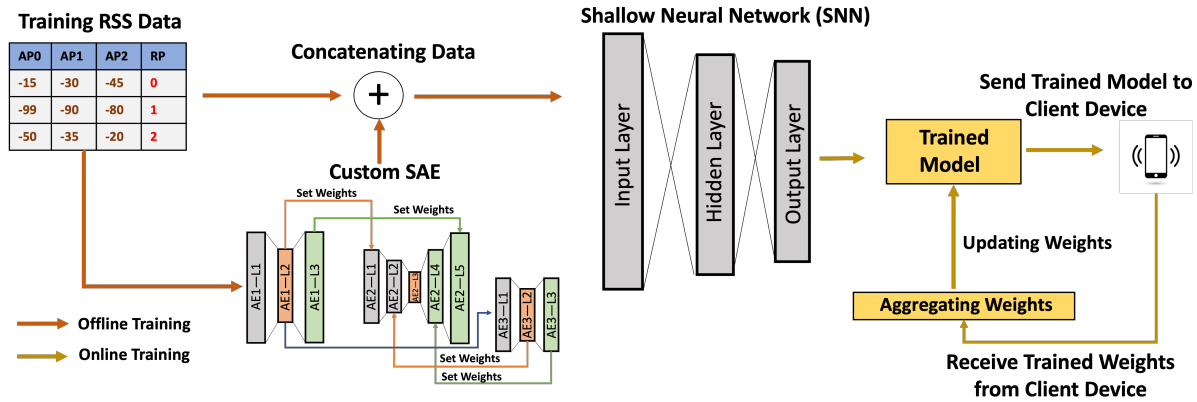


Figure 47: GM architecture for FedHIL

The offline training phase consists of two components: A data augmentation strategy using a custom stacked autoencoder (SAE) and a shallow neural network (SNN) model. Data augmentation using a custom SAE is a key component of the proposed framework. The custom SAE learns to extract meaningful features from the input RSS data and generates augmented data that can help improve the accuracy of the model. The output from the custom SAE is concatenated with the original RSS data and then sent for training the SNN model. The SNN model is chosen to have a lightweight architecture, which makes it well suited for mobile devices. Once the SNN model is trained, it is made available as the GM. The offline training phase is complete at this point, and the trained GM is then made available for deployment on client for prediction.

In the online training phase, the client device makes a copy of the GM and stores it locally. The client device uses the trained GM, now called the LM, to make predictions for the user with an unknown location. While this prediction is happening, the client device also retrains the LM using its local data. The updated trained weights in the LM are then sent back to the server, which holds the GM. The server receives the updated weights only and aggregates them with the GM, thus updating the GM without looking at the raw RSS data from the client devices. This process enables the GM to adapt to new heterogeneous data from multiple clients while maintaining the privacy of the client devices.

6.6.2. CUSTOM STACKED AUTOENCODER (SAE) NEURAL NETWORK

To enhance the collected RSS data prior to training, we customize an SAE neural network model to create augmented RSS data. Our SAE model comprises of multiple autoencoders (AE) that are trained one layer at a time using a layer-by-layer training approach. The AE neural network consists of three key components: an Encoder, a Decoder, and a Bottleneck layer. Each of these components is comprised of one or more fully connected layers with varying numbers of neurons. The encoder layers are responsible for compressing the input data and reducing its dimensionality, while the decoder layers are utilized to reconstruct a noisy representation of the input data back to its original dimension. The bottleneck layer represents the input data at its lowest dimension and reflects the most salient and significant features in the data.

The encoder layers consist of fully connected layers with decreasing numbers of neurons in each layer. These layers perform the compression and dimensionality reduction operations. By reducing the number of neurons in each layer, the encoder layers effectively reduce the dimensionality of the input data. Non-linear activation functions, such as ReLU (Rectified Linear

Unit), are used in these layers to introduce non-linearity and capture complex relationships in the data. The bottleneck layer is positioned at the center of the AE and has the lowest dimensionality among all the layers in the network. It acts as the bottleneck for information flow, representing the most salient and significant features of the input data. The bottleneck layer typically has the least number of neurons, allowing it to capture the essential information while discarding less relevant or redundant information. The reduced dimensionality in this layer helps in extracting the most discriminative features from the input data. The encoder can be considered a neural network that transforms high-dimensional input data into lower-dimensional data (at the bottleneck layer).

The decoder layers mirror the structure of the encoder layers but in reverse order. They consist of fully connected layers with increasing numbers of neurons in each layer. These layers are responsible for reconstructing the compressed representation back to the original dimension. By increasing the number of neurons in each layer, the decoder layers expand the compressed representation and attempt to reconstruct the input data. The decoder layers are designed to reconstruct the data in the same way it was reduced, ensuring the preservation of relevant information and the underlying structure of the data. The layer-by-layer training approach employed in the SAE model is specifically designed to ensure minimal noise during the reconstruction process in the decoder. This approach allows each layer to learn an effective representation of the data before proceeding to the next layer. By gradually building up the complexity of the reconstruction, the network can better capture important details and relationships within the data. This results in more stable training and better overall performance of the model.

The proposed SAE model involves stacking three AEs together, each having its own fully connected layers, as shown in Figure 47. AE1 and AE3 are trained layer-wise, with the output of the second layer (L2) of AE1 being fed as input to AE3. AE2 is positioned as illustrated in Figure

47 and receives the trained weights from AE1 and AE3. The SAE model is trained layer-by-layer to minimize the reconstruction error between the input and output data. The advantage of this approach is that it allows the network to learn an effective representation of the input data at each layer before proceeding to the next one, rather than attempting to train all the layers simultaneously. This can result in a more stable training process and better overall performance.

The SAE model is leveraged for RSS data augmentation by training it on the training RSS data created during the offline phase. The SAE can subsequently generate synthetic RSS measurements that can be used to augment the training dataset. The synthetic RSS data emulates the variations that can occur across different devices, potentially improving the system's robustness and generalization by providing a more diverse set of training data that accurately represents variations across different devices.

6.6.3. SHALLOW NEURAL NETWORK (SNN)

The SNN model is an artificial neural network that receives concatenated input from both the training and augmented RSS data. Our SNN model is comprised of three fully connected layers, namely, the input layer, the hidden layer, and the output layer. The input layer is responsible for receiving the concatenated input data, which is then passed on to the hidden layer. The hidden layer is where the complex processing of the input data occurs, using various activation functions, such as rectified linear unit (Relu), and weights assigned to each neuron in the layer. The output of the hidden layer is then passed on to the output layer, which provides the final result of the model's prediction. One advantage of our SNN model is that it is lightweight (it has only a single hidden layer), making it easy to deploy on client devices. This is particularly beneficial for indoor localization as we require real-time prediction and processing of data, and a lightweight model

minimizes the computational resources required. Furthermore, the augmented data used in the training of the SNN model allows it to generalize for device heterogeneity during indoor localization. This means that the model is able to recognize and predict the location of a user, even if they are using different devices to access the application.

6.6.4. LOCAL MODEL (LM) ARCHITECTURE

The client device receives the trained GM from the central server, as illustrated in Figure 48. Once received, the client device clones the trained GM and stores it locally as the LM, which is used to make predictions.

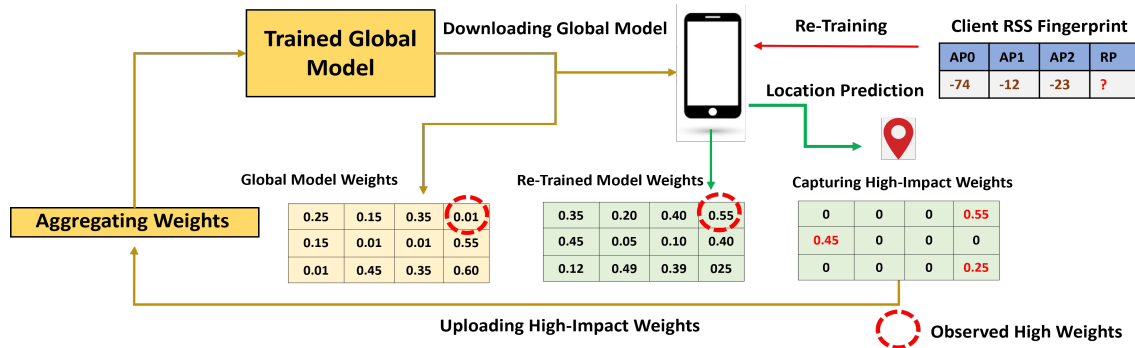


Figure 48: LM architecture for FedHIL

To enhance the accuracy of predictions, the client device scans for nearby Wi-Fi APs and stores their RSS data locally. This stored RSS data is used to inference with the LM and to retrain the LM to better fit the user's environment. This local training is supervised, and the client device uses its stored data to update its own local model through backpropagation with stochastic gradient descent. Gradients are calculated based on the loss function of the local model. The LM is retrained over a small number of epochs to ensure that the process has low overhead on the resource-constrained client device. After the LM is retrained, the client device uses an absolute distance

method, which is part of the FedHIL aggregation technique (discussed in Section 6.6.3), to select only a subset of LM weights to upload to the server for aggregation. This is done to protect the GM from noisy, low impacting weights. By only transmitting a high impact subset of weights, not only are the communication requirements for the FL process relaxed, but overall accuracy can also be improved. Throughout this process, multiple client devices participate in the prediction and contribute to updating the GM with new heterogeneous device-specific RSS data. Figure 48 illustrates the selection process of the high-impact weights using the absolute distance method, which is discussed in more detail in the following subsection.

6.6.5. FEDHIL AGGREGATION TECHNIQUE

We now describe our FL aggregation method that has been specifically designed to address the challenges of heterogeneity and long-term accuracy robustness in indoor localization. Traditional FL techniques may cause the ML model to degrade in accuracy over multiple rounds of client-side retraining, due to noisy data collected from the client devices. These outliers in the online training phase can skew the ML model’s performance and cause it to degrade. Our FedHIL aggregation technique addresses this issue by identifying selected features in the trained LM of the client during the online training phase. The selected features are the high-impacting features in the locally collected RSS fingerprints of the client device, calculated by finding the absolute distance of the weights between the trained LM and the cloned GM that each client maintains before retraining it. This absolute distance represents the changes in the weights for the visible features in the LM. Our technique then filters these absolute weights by only capturing the top $H\%$ of the highest changes in the weights. This means that the top $H\%$ of the largest magnitude difference between the LM and GM are captured, and the background noise and potentially noisy distribution

in the weights picked up by the client device are filtered out. The top $H\%$ of weights from the client are then sent back to the server for aggregation. The received weights from the client device are then averaged with the existing GM in the server via simple averaging. This process is repeated with participating client devices, with each device contributing to the improvement of the ML model's performance. The equations below show a mathematical representation of the proposed technique:

$$W_{client} = W_T - \eta * \nabla lf(W_T) \quad (24)$$

This equation is the same as in FedAvg, where W_{client} are the weights generated from the LM using SGD, W_T are the weights from the GM at the beginning of the round, η is the learning rate, and $\nabla lf(W_T)$ is the gradient of the loss function with respect to the GM weights.

$$W_{abs} = |W_{global} - W_{client}| \quad (25)$$

$$I_{High} = \psi(W_{abs}, H) \quad (26)$$

Equations 25, and 26 identify the high-impact weights in the W_{client} from Equation 26. Equation 25 involves calculating the absolute difference between the weights in the GM (W_{global}) and the weights of the LM (W_{client}). By calculating the absolute difference, we can identify the areas where the GM and LM differ the most. Equation 27 involves determining the indices of the top $H\%$ of weights. The variable $H\%$ represents the top percentage of the largest magnitude

difference between the GM and the LM. The ψ function is used in selecting the indices associated with the top $H\%$ weights in W_{abs} .

$$W_{High} = \begin{cases} W_{client}, & \text{if } I_{client} \in I_{High}, \\ 0, & \text{otherwise} \end{cases} \quad (27)$$

In Equation 27, we capture the high impact weights in W_{client} by setting the non-high-impact weights to 0 and keeping the high-impact weights the same as before. This is done to reduce the impact of client-to-client variation on the high-impact weights.

$$W_{new\ global} = \frac{1}{N} \sum_{i=1}^N W_{high} + W_{global} \quad (28)$$

Finally, Equation 28 computes the new GM weights by taking the average of the selected high impact client model weights (W_{high}) and the original GM weights (W_{global}), where N is the number of clients.

By following this technique, we update the GM with only high-impacting weights from the client devices. The proposed technique has several advantages. First, the high-impacting weights can prevent the ML model from overfitting, thereby preventing the GM model from degrading. Second, by only capturing the top $H\%$ of the weights, the proposed technique filters out noise, leading to better performance of the ML model. Third, by having multiple clients with heterogeneous data participating in this process, the overall system can achieve heterogeneity

resilience. Lastly, since only a subset of the trained weights is shared, compression techniques can compress sparse (zero) weights to reduce the wireless communication volume and latency from the client devices to the server. These factors make *FedHIL* a promising FL approach for indoor localization.

6.7. EXPERIMENTAL SETUP

In this section, we present the experimental setup for the evaluation of the proposed FedHIL framework. The aim of our experiments is to evaluate the performance of the FedHIL framework in real-world settings and compare it with other FL and non-FL based indoor localization frameworks.

Table 8: Trainable parameters in each layer in the custom SAE.

AE1-L1	AE1-L2	AE1-L3	AE2-L1	AE2-L2	AE2-L3	AE2-L4	AE2-L5	AE3-L1	AE3-L2	AE3-L3	Total
14878	10414	14964	10414	5103	3570	7388	14964	7289	5103	7388	101475

In the FedHIL framework, we pre-train the GM using RSS data collected from a single smartphone. This is done to ensure that the framework is resilient towards heterogeneity and can be evaluated on any number of smartphones. To pre-train the GM, we first augment the original RSS data using a custom SAE, with the trainable parameters shown in Table 8. The custom SAE uses SGD for optimization with the MSE (mean square error) loss function. The custom SAE is trained for 700 epochs per layer, to generate the best results for the given data. We then concatenate the original and augmented data and use it to train the SNN model. The proposed SNN model in the FedHIL framework has a Relu non-linearity with 256 neurons in the hidden layer and uses the sigmoid function in the output layer for prediction. The SNN also uses the SGD optimization with

the Sparse Categorical Crossentropy loss function and trained for 1200 epochs. The SNN model is trained with the trainable parameters shown in Table 9.

Table 9: Trainable parameters in each layer in the SNN model.

Input Layer	Hidden Layer	Output Layer	Total
22144	33024	15677	70845

After pre-training the GM, we make it available to each client device for testing and retraining. The client devices retrain the SNN model over 10 epochs during the online training phase, to fit the resource constraints in the client devices and report only the top H% of the high impacting features in the local model back to the server. We set the H parameter in the proposed FedHIL aggregation technique based on our sensitivity analysis presented in Section 6.7.2. To evaluate the performance of the FedHIL framework, we measured the localization accuracy using the Euclidean distance formula:

$$Euclidean\ Distance = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2 + (Z_2 - Z_1)^2} \quad (44)$$

where (X_1, Y_1, Z_1) represents the ground truth location coordinates and (X_2, Y_2, Z_2) represents the predicted location coordinates. The accuracy is thus calculated based on the distance between the ground truth location and the estimated location of a device, which is based on the RSS data collected by the device. This approach allows us to assess the accuracy of the framework across multiple devices and scenarios, ensuring that the framework can be reliably used in real-world settings.

We compared the performance of our FedHIL framework with three FL-based frameworks (CRNP [164], FedAeDNN [165], and FedLoc [166]) and four non-FL-based frameworks (ANVIL

[163], SHERPA [46], VITAL [80] and WiDeep [17]). The FL-based frameworks were selected because they have shown promising results in addressing device heterogeneity, which is a key challenge. On the other hand, we chose the non-FL-based frameworks to provide a baseline for comparison and to highlight the improvements achieved using FL.

Table 10: Details of smartphones used in our experiments.

Manufacturer	Model	Abbreviation	Wi-Fi Chipset
BLU	Vivo 8	BLU	MediaTek Helio P10
HTC	U11	HTC	Qualcomm Snapdragon 835
Samsung	Galaxy S7	S7	Qualcomm Snapdragon 820
LG	V20	LG	Qualcomm Snapdragon 820
Motorola	Z2	MOTO	Qualcomm Snapdragon 835
Oneplus	Oneplus 3	OP3	Qualcomm Snapdragon 820

To collect data for our indoor localization experiments, we utilized six smartphones from different manufacturers, each equipped with a different Wi-Fi chipset, as shown in Table 10. We normalized the RSS values between 0 and 1 to train the ML model on a consistent and uniform dataset, which can lead to better performance and more accurate indoor localization.

We select five different buildings (Building 1-5) with different salient features. and performed experiments on paths of different shapes and with lengths ranging from 60 to 88 meters. Building 1 is made of wood, and concrete, and is surrounded by computers with large open areas. A total of 78 unique Wi-Fi APs were visible in this building. Building 2 and Building 3 are relatively new buildings with a mix of metal and wooden structures, open areas, and bookshelves with heavy metallic equipment. We observed 218 and 112 unique APs, respectively in these buildings. Building 1 and Building 2 are located in close proximity to each other and are separated by a single wall. The floorplans in these two buildings were chosen to consider the impact of interference from different sources in the two buildings. The mix of different materials and equipment in the

buildings creates a complex indoor environment, which, together with interference from Wi-Fi APs in the other building, poses a challenge for indoor localization systems. Building 4 has several small offices, with a mix of metal, wood, and glass structures. The building was densely covered by 156 APs. Building 5 has laboratory equipment with a considerable amount of electronic and mechanical equipment and 125 visible APs.

The presence of metal and electronics in the vicinity of the tested paths in the buildings lead to noisy Wi-Fi fingerprints that complicate indoor localization efforts. We captured Wi-Fi RSS fingerprints during regular working hours, without artificially manipulating human occupancy. By collecting Wi-Fi RSS fingerprints during the course of entire days, including regular working hours, we were able to capture various forms of interference in Wi-Fi signal propagation. Our data collection methodology allowed us to incorporate various sources of interference, including dynamic factors such as population density and human movement within the building. By capturing data in this authentic setting, our aim is to gain a comprehensive understanding of the challenges and complexities encountered in real-world indoor localization, thereby enhancing the generalizability of our results. For the offline phase, we used five fingerprints per RP to train the GM, and for the online phase, we used one fingerprint per RP. We selected a granularity of 1-meter separation between RPs for our experiments, which we consider sufficient for localizing humans.

Table 11: Inference time latency for the SNN model on different smartphones in milliseconds (ms).

BLU	HTC	LG	MOTO	OP3	S7	Average
73.35 ms	62.05 ms	64.81 ms	76.87 ms	64.16 ms	67.25 ms	68.08 ms

To assess the lightweight nature of the SNN model implemented on the smartphones listed in Table 10, we measured the inference time latencies for each device using the SNN model during the online phase, as shown in Table 11. The inference speed is influenced by various smartphone resources, including processor speed, memory, and other hardware capabilities. Based on our measurements, the average inference time latency across all smartphones was found to be 68.08 ms, indicating the efficient and lightweight performance of the SNN model on these devices. In the following subsections, we present sensitivity analysis, comparison of the proposed FedHIL framework with state-of-the-art indoor localization frameworks, and analysis for skew and scalability.

6.7.1. SENSITIVITY ANALYSIS FOR FEDHIL

In this section, we analyze the H parameter value within FedHIL and evaluate the framework's performance on real-world data. The parameter determines percentage of the trained weight values in the LM to be aggregated with the GM. Figure 49 shows experimental results for selecting the H parameter, whose value is shown ranging from 10% to 100%. The results shown are for the Building 5 floorplan (highly noisy environment) with the MOTO device used for training and OP3 device used for testing in the online phase. From the Figure it can be observed that setting H to 10% causes the model to overfit easily, leading to a significant degradation in localization error (y axis), particularly after RP 5, as only a tiny proportion of the trained weights in the client LM are used to update the GM. Additionally, the model is heavily affected by noise, resulting in an increase in localization errors. Setting H to 20% yields the best results, where the model is most resilient towards noisy RP locations. Although the model with $H=20\%$ shows a few instances of high errors, they are much smaller than those observed for other H values. The H values from 30%

to 100% show similar results, demonstrating some resilience towards noise, but they are still heavily affected by it, resulting in higher localization errors compared to the 20% value. The 100% H value is where all the trained LM weights are averaged with the GM, which is the same as the FedAvg aggregation approach. The approach shows high localization errors and is heavily affected by noise.

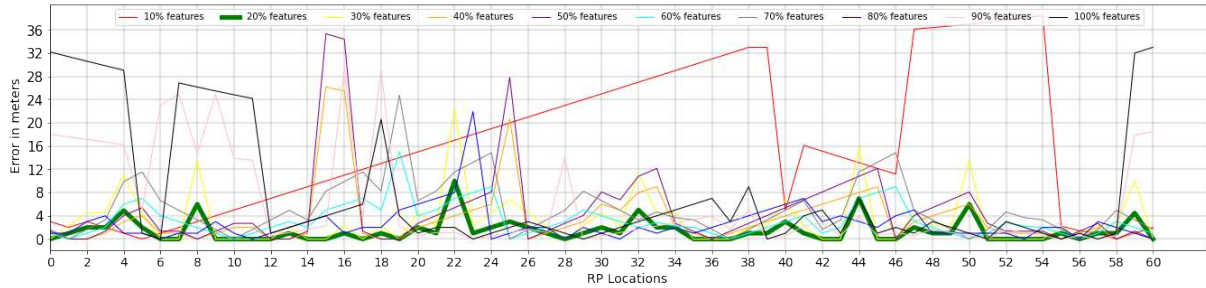


Figure 49: Sensitivity Analysis of RP Locations in Building 5 Floorplan for Selecting H Value in FedHIL Technique

To thoroughly analyze the impact of the H parameter value within the FedHIL framework and determine the optimal H value, we conducted experiments and plotted the average error for different H values. In Figure 50, we present the results, where each H value ranges from 10% to 100% with a step size of 10. Upon analyzing the plot, several observations can be made. Firstly, setting H to 10% resulted in the highest average error compared to other H values. This can be attributed to a significant loss of weights during aggregation, leading the model to easily overfit the data.

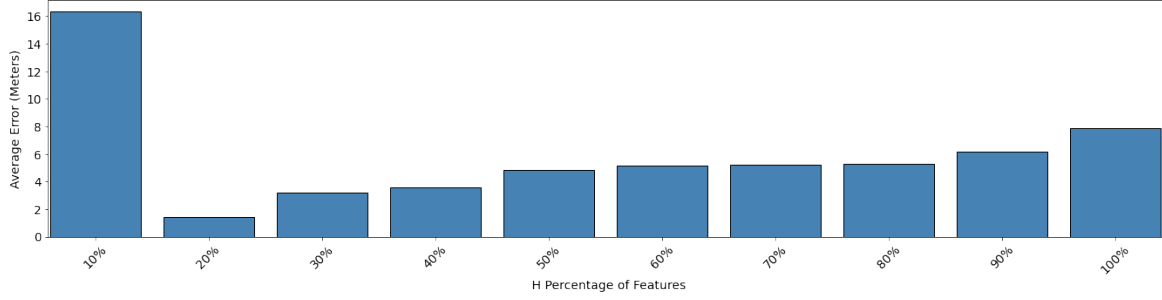


Figure 50: Average error in meters for different H value in FedHIL technique.

On the other hand, when we set H to 20%, we observed the lowest average error. This indicates that preserving 20% of the highest changing weights during aggregation strikes a balance between retaining important information and avoiding overfitting. As a result, the model achieves better generalization and performs more accurately. For H values of 30% and 40%, we observed slightly higher errors compared to H = 20%, but significantly lower errors than H = 10%. This suggests that these H values strike a trade-off between preserving important information and allowing for some level of noise or variability in the aggregated weights. Furthermore, H values of 50%, 60%, 70%, 80%, and 90% showed similar average errors, albeit slightly higher than H = 30% and H = 40%. These values still maintain a good level of performance by preserving a considerable portion of the highest changing weights during aggregation. Lastly, when setting H to 100%, we observed the highest error among the range of 20% to 90%. Nevertheless, even H = 100% showed a significantly lower error compared to H = 10%, indicating that preserving all weights during aggregation still yields improvements over discarding all weights. Based on these findings, we conclude that setting H to 20% provides the optimal balance between weight preservation and error reduction in our experiments, even in the presence of high noise in the floorplan (Building 5). We ultimately set the H parameter in FedHIL to 20%, as it also provided the best performance across other buildings. We used this value for FedHIL in all our subsequent experiments.

6.7.2. CLIENT-SERVER COMMUNICATION LATENCY

In this section, we analyze the communication latency between client devices and the server in our FedHIL framework. Figure 51 shows the average communication latency for different H values across all RPs in Building 5. We observe that as the H value increases, the communication latency also increases. This is because a higher H value involves transmitting more data from the client to the server, resulting in longer transfer times. However, the average communication latency differences between H values are relatively small, always remaining below 1 second. Analyzing and optimizing communication latency is crucial in FL as it directly impacts system performance and efficiency.

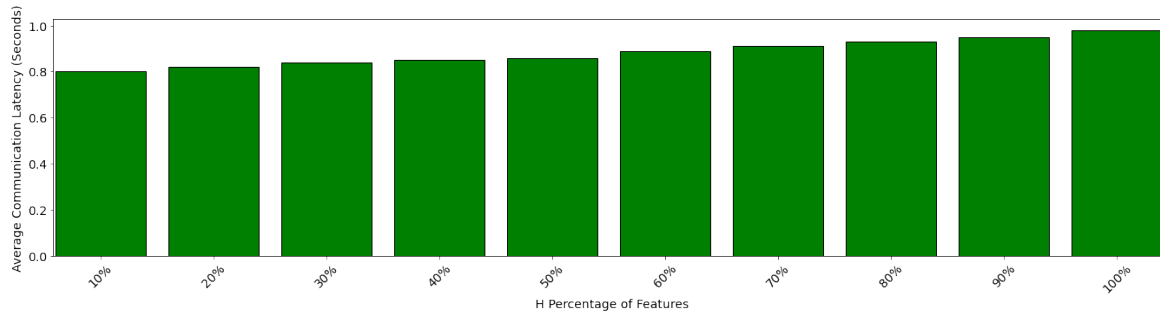


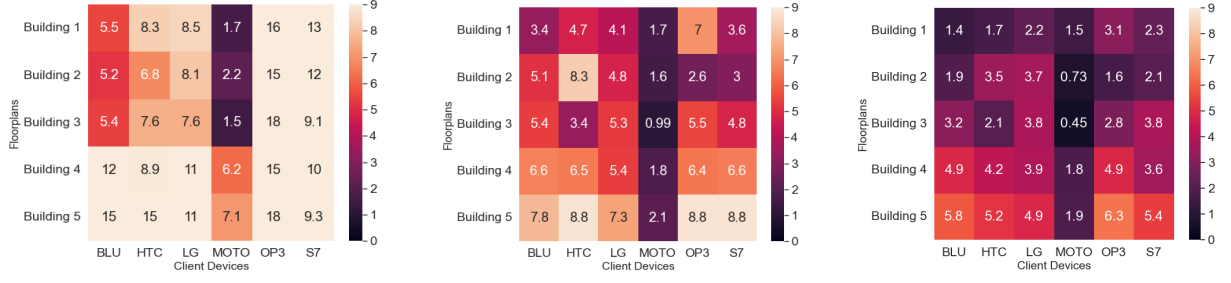
Figure 51: Average communication latency in seconds for different H value in FedHIL technique.

A higher communication latency can result in delays in aggregating the model updates, leading to slower convergence and overall training performance. On the other hand, minimizing communication latency enables faster synchronization, allowing the system to adapt and learn more quickly. Combining the findings from Figure 51, which illustrates the relationship between H values and average error, we observe that $H = 20\%$ achieves the lowest average error, indicating better localization accuracy. Additionally, $H = 10\%$ exhibits the lowest communication latency,

but it compromises performance. Higher H values show slightly higher communication latency without significant improvements in accuracy. Considering these aspects, we conclude that H = 20% strikes a balance between performance and communication latency. It offers the optimal solution within the FedHIL framework, ensuring accurate localization while maintaining efficient communication.

6.7.3. IMPACT OF CUSTOM SAE IN FEDHIL

In order to evaluate the impact of the custom SAE in addressing device heterogeneity challenges during indoor localization, we conducted a series of experiments comparing different configurations of the FedHIL framework. Specifically, we examined three situations during the offline training phase: FedHIL without SAE (no augmentation), FedHIL with a traditional SAE, and FedHIL with the custom SAE. To introduce data augmentation, we incorporated a traditional SAE into the FedHIL framework. The parameters of this traditional SAE were set to be identical to those of the custom SAE. However, it is important to note that the traditional SAE was not trained using a layer-by-layer approach, which is a key characteristic of our custom SAE. To assess the performance of the trained SNN model, we conducted experiments using MOTO as the training device. We compared the results obtained from these experiments with the three previously mentioned situations, as illustrated in Figure 52. This comparison allowed us to analyze the influence of the custom SAE on the accuracy and effectiveness of the FedHIL framework in addressing device heterogeneity challenges during the offline training phase.



(a) FedHIL without SAE (b) FedHIL with traditional SAE (c) FedHIL with custom SAE

Figure 52: FedHIL with different SAE configurations, training device: MOTO (values in cells represent Euclidean distance).

Figure 52 (a) illustrates the outcome of FedHIL without the SAE, Figure 52 (b) depicts the results with the traditional SAE, and Figure 52 (c) depicts the results with our proposed custom SAE. The evaluation aimed to assess the impact of the custom SAE on the localization performance across various client devices and building floorplans. To conduct the evaluation, we utilized the dataset consisting of six client devices and five different building floorplans, as discussed in section 6.7.1. For each combination of the client device and building floorplan, we measured the localization error using the Euclidean distance method. The localization error represents the discrepancy between the estimated position and the ground truth position of the client device. In Figure 52, each cell corresponds to a specific combination of client devices and building floorplan. The value displayed within each cell represents the average localization error observed across all available RPs within the corresponding building floorplan.

By analyzing these Figures, we can clearly observe the importance of our proposed custom SAE in our FedHIL framework. The results indicate that the underlying SNN model without SAE only performs well in the absence of device heterogeneity. This can be observed from the lower localization errors for MOTO in the online phase, which is the same device that is used to train the model in the offline phase in Figure 52 (a). Without SAE, the model fails to generalize on

heterogeneous devices other than MOTO, including LG, OP3, and S7 (performance for BLU and HTC devices is slightly better, but still poor), and also in high noise environments, such as in Buildings 4 and 5. However, with the addition of the traditional SAE, we observe significant improvements in localization errors compared to those without using the SAE. The incorporation of the traditional SAE allows for some level of adaptation to device heterogeneity, leading to better performance across different devices and noise environments. This is evident from the reduced errors in the localization, especially for devices other than MOTO. However, it is important to note that the performance improvement is limited, and the model still struggles to achieve satisfactory accuracy in challenging scenarios, such as Building 4 and Building 5. On the other hand, with the custom SAE, we witness even more remarkable improvements in localization errors, surpassing the performance achieved with the traditional SAE. The custom SAE, trained in a layer-by-layer approach specifically tailored for our domain, effectively addresses the challenges posed by device heterogeneity. By augmenting the training data and diversifying its distribution, the custom SAE enables the SNN model to adapt and generalize across a wide range of devices and noisy environments.

The substantial reduction in localization errors with the custom SAE demonstrates its superior ability to enhance the robustness and accuracy of the FedHIL framework. The model now achieves reliable and accurate indoor localization across various devices and noise levels, indicating the effectiveness of our approach in mitigating the impact of device heterogeneity.

To summarize the effectiveness of the custom SAE within the FedHIL framework, we present a bar plot to compare the average error in meters across the three previously mentioned scenarios in Figure 53. The experimental setup aligns with the conditions in Figure 53.

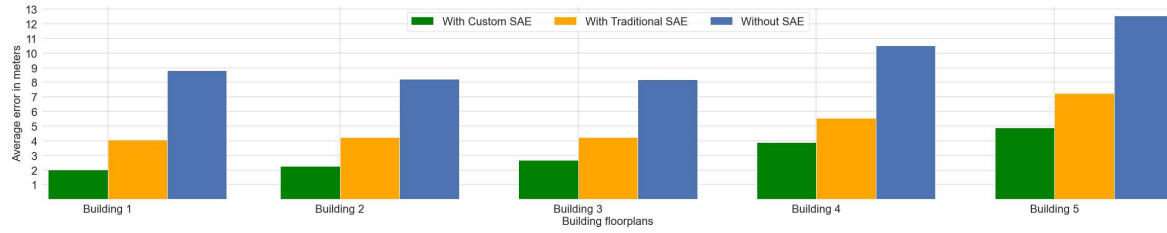


Figure 53: Impact of Custom SAE on Average Error in Building Floorplans

Figure 53 serves as compelling evidence showcasing the substantial positive impact of the custom SAE on the FedHIL framework's ability to handle device heterogeneity. The bar plot illustrates the average error per building floorplan, providing valuable insights into the framework's performance across different scenarios. The results are striking, revealing significant reductions in average errors when utilizing the custom SAE. Building 1 demonstrates a remarkable 77.1% reduction in errors compared to the scenario without SAE, and a noteworthy 50.2% error reduction compared to the traditional SAE. Building 2 follows closely, with a reduction of 72.59% compared to without SAE and a substantial 46.8% error reduction compared to the traditional SAE. Building 3 showcases a 67.19% error reduction compared to without SAE, and a commendable 36.4% error reduction compared to the traditional SAE. Building 4 exhibits a 63.08% reduction in errors compared to without SAE, and a notable 30.09% error reduction compared to the traditional SAE. Lastly, building 5 experiences a remarkable 60.9% reduction in errors compared to without SAE, along with a significant 32.36% error reduction compared to the traditional SAE. These findings powerfully emphasize the efficacy of the custom SAE in enhancing the framework's resilience to device heterogeneity, resulting in improved accuracy in indoor localization across diverse buildings. By effectively addressing the unique characteristics of each device, the integration of the custom SAE empowers the FedHIL framework to deliver more reliable and accurate localization results.

To provide a clear analysis of the impact of the custom SAE, we present a plot in Figure 54 that compares the original fingerprint with the augmented fingerprint generated using the custom SAE. Upon careful examination, we observe distinct variations in the augmented fingerprint compared to the original fingerprint. The custom SAE effectively captures and incorporates device heterogeneity, resulting in a more diverse and representative fingerprint. This augmentation process enables the model to learn and adapt to the unique characteristics of different devices, enhancing its ability to handle device heterogeneity. The differences between the original and augmented fingerprints are indicative of the custom SAE's capability to capture important features and patterns that can potentially arise due to device heterogeneity. These additional features contribute to a more comprehensive and accurate representation of the indoor environment, leading to improved localization performance.

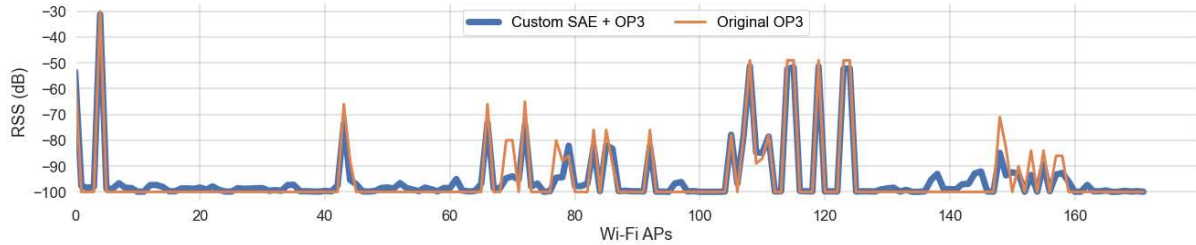


Figure 54: Comparison of Original Fingerprint and Augmented Fingerprint Using Custom SAE on Device: OP3

6.7.4. COMPARISON WITH PRIOR FL AND NON-FL INDOOR LOCALIZATION FRAMEWORKS

In this section, we present experimental results of comparing FedHIL with other state-of-the-art frameworks. We begin by comparing the performance of FedHIL with FL and non-FL based frameworks that have been specifically curated for device heterogeneity. The FL frameworks

evaluated are FedHIL, CRNP [164], FedAeDNN [165], and FedLoc [166], while the non-FL frameworks are ANVIL [161], SHERPA [46], VITAL [80], and WiDeep [17]. Figure 55 shows box plot results from this comparison study. The presented data in the Figure is the average error of each framework with all testing client devices across all building paths.

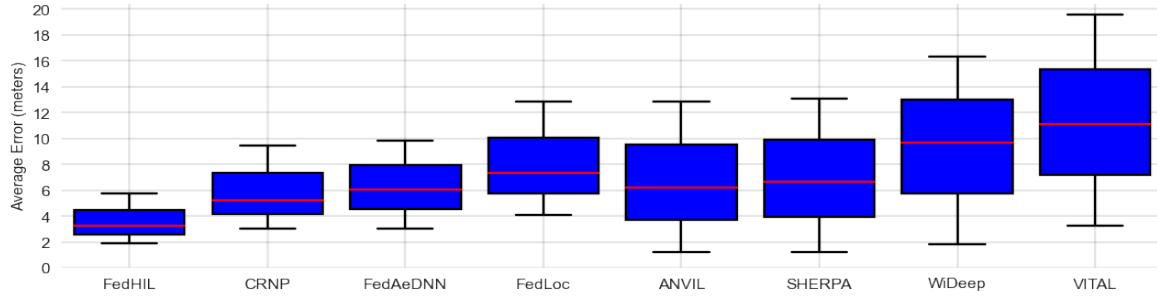
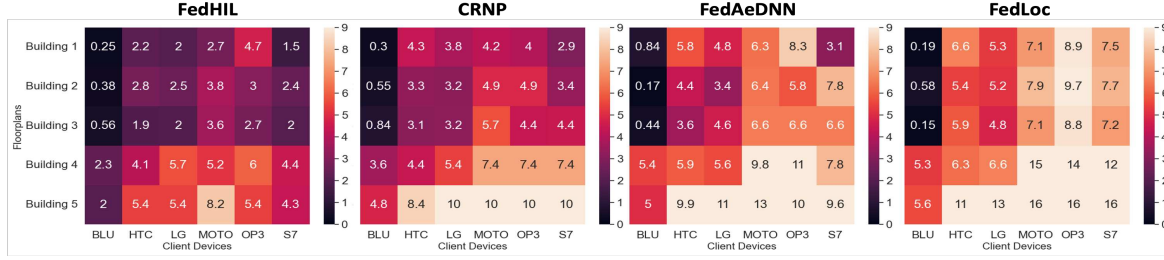


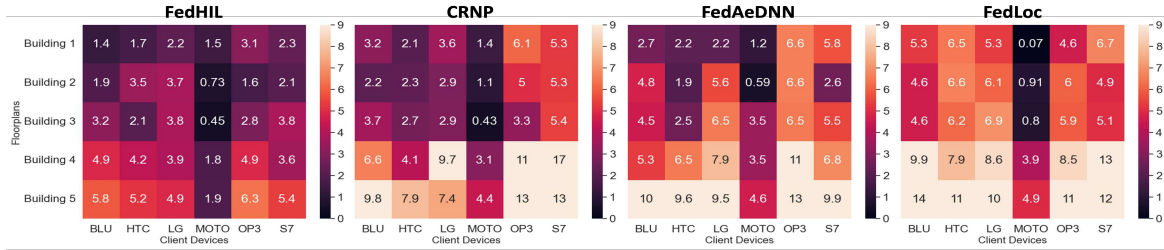
Figure 55: Average localization across all smartphones and floorplans for comparison with FL and non-FL frameworks

It can be observed that the FL frameworks demonstrate superior results compared to the non-FL frameworks in terms of average localization error. This indicates that FL, which leverages multiple devices to train a model collectively, is more effective in addressing device heterogeneity, resulting in more accurate localization. FedHIL outperforms all other FL (and non-FL) frameworks, with the lowest average error of 3.24 meters. Specifically, compared to the FL-based indoor localization frameworks CRNP, FedAeDNN, and FedLoc, our FedHIL framework is able to achieve 1.62 $\times$, 1.86 $\times$, and 2.26 $\times$ lower average localization error, respectively. The non-FL frameworks ANVIL and SHERPA perform better than FedLoc, but still have higher average errors compared to the other FL frameworks. ANVIL and SHERPA use more powerful ML models such as attention networks and KNN with DNN, which explains their better performance. However, the FL frameworks use simpler DNN models and still outperform the complex non-FL frameworks. Analyzing the minimum and maximum errors of each framework, FedHIL has the lowest

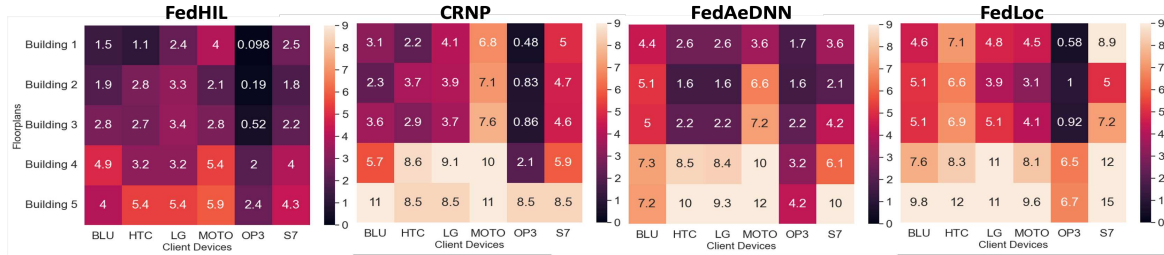
minimum and maximum error among all the frameworks. CRNP, FedAeDNN, and FedLoc have similar minimum and maximum errors. Among the non-FL frameworks, ANVIL and SHERPA have the lowest minimum errors, but their maximum errors are higher than all FL frameworks excluding FedLoc. We observe that WiDeep and VITAL show higher errors compared to other frameworks as these models use GPC (WiDeep) and vision transformers (VITAL) to perform their computation. These models are larger, more complex, and require huge amounts of data for effective performance, indicating the need for a larger database of heterogeneous data to effectively utilize the potential of their underlying models.



(a): Device-specific performance across all building floorplans and FL frameworks with training device: BLU



(b): Device-specific performance across all building floorplans and FL frameworks with training device: MOTO



(c): Device-specific performance across all building floorplans and FL frameworks with training device: OP3

Figure 56: Device- specific performance across all building floorplans and FL frameworks for different training devices.

To better understand the performance of the FL frameworks, we present a detailed heatmap view that shows the average error values for each device across the five buildings. In Figure 56, each box indicates the average error for each building floorplan and smartphone. In this analysis, we present results for three training devices. In Figure 56 (a), we trained all the frameworks on BLU, Figure 56 (b) on MOTO, and Figure 56 (c) on OP3. We omit the results from training on the

remaining three devices for brevity (trends were similar to those observed in Figure 56). From Figure 56, it can be seen that FedLoc performs well on floorplans with less noise (Building 1, 2, and 3) but fails to provide accurate results on high noise floorplans (Building 4 and 5), even for the same training device that the model was trained on. FedAeDNN has better performance than FedLoc, but still fails to provide accurate results in Building 4 and Building 5. However, it significantly reduces errors from FedLoc, indicating greater resilience towards heterogeneity. CNRP outperforms FedLoc and FedAeDNN since the SAE used in it is better suited for heterogeneity. The FedAvg aggregation approach, used in CNRP and FedAeDNN, performs better than FedLoc's FedSGD aggregation approach. The weighted averaging of FedAvg prevents overfitting, while FedSGD overfits due to gradient aggregation, making it more appropriate for less noisy environments. FedHIL exhibits the lowest localization errors for each device compared to other FL-based frameworks. It provides accurate results even in floorplans with high noise (Building 4 and 5), effectively mitigating bottlenecks observed in other frameworks (e.g., in Figure 56 (b) see: Building 1-OP3/S7, Building 2-OP3/S7, Building 3-S7, Building 4-LG/OP3/S7, and Building 5-BLU/HTC/LG/MOTO/OP3/S7).

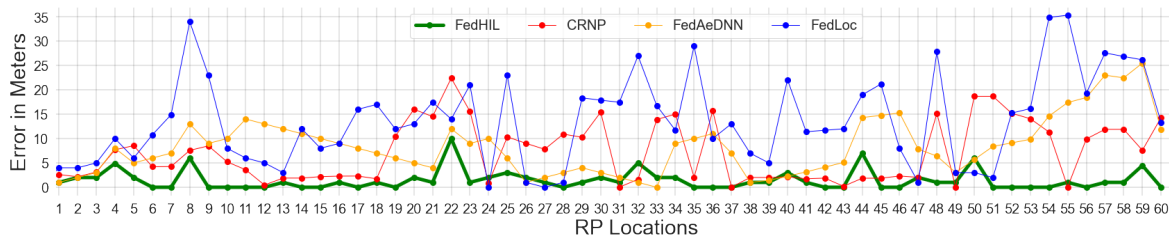


Figure 57: Susceptibility to noise across FL frameworks with training device : MOTO, testing device OP3 in Building 5.

Figure 57 depicts the susceptibility of various FL-based frameworks to noisy data collected in the online phase at each RP, including FedHIL, in the high-noise environment of Building 5 with

MOTO as the training device and OP3 as the testing device. The noisy environment in Building 5 led to many instances of introducing noisy or outlier fingerprints, leading to the training of the ML model with noisy data, which can degrade the performance of FL-based frameworks. Our FedHIL framework and its selective aggregation technique, designed to prevent the underlying SNN model from degrading due to noise, outperforms other FL frameworks. The combination of the proposed SAE and the novel aggregation technique reduces the error even on highly noisy RPs, demonstrating the superior performance of FedHIL in noisy environments.

6.7.5. SKEWNESS ANALYSIS

A skewness analysis was conducted using six different users, each with a smartphone (client). The client device combinations were split into five cases, with each case consisting of a different combination of device types. All the cases were initially trained on the MOTO device and evaluated on all the building floorplans. In Figure 58, we plot the average localization error per case per floorplan with all device combinations in the case split. The device combinations in each of the cases are as follows : Skew Case 1 contains two unique devices (5 users with BLU and 1 with S7), Skew Case 2 contains three unique devices (4 users with BLU, 1 with OP3, and 1 with S7), Skew Case 3 contains four unique devices (3 users with BLU, 1 with MOTO, 1 with OP3, and 1 with S7), Skew Case 4 contains five unique devices (2 users with BLU, 1 with LG, 1 with MOTO, 1 with OP3, and 1 with S7), and Skew Case 5 contains six unique devices (each user has a different device: BLU, MOTO, LG, OP3, S7, and HTC).

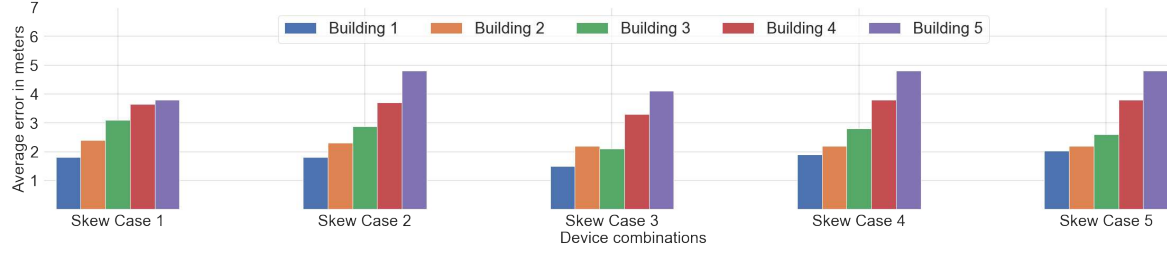


Figure 58: Skewness analysis on different floorplans and devices with different device mixes among clients.

The results in Figure 58 indicate that the FedHIL framework is fairly resilient to device skewness. Even in the presence of skewness, the model performance remains relatively stable across scenarios (comparing for the same building floorplan). In Skew Case 1, we observe lower localization errors in all buildings. This is because in the online training phase, the data is weighted more for a particular device, and hence the model learns to generalize better, even when there is one other heterogeneous device. In Skew Case 2, we observe similar results as that of Skew Case 1, but with a slight error increase for Building 5. This is due to the high-noise environment in that building. The Skew Case 3 shows similar results as for Skew Case 1 and Skew Case 2. The Skew Case 4 and Skew Case 5 scenarios show similar results even in high noise environments but show slightly higher errors compared to Skew Case 3, due to the added diversity in the participating client devices. Although Skew Cases 4 and 5 contain a greater number of heterogeneous client devices, they show similar performance as in Skew Case 2. From the results, we can conclude that the FedHIL framework shows resilience to device skewness, making it practical for deployment in real-world scenarios where it can provide good accuracy even when the device heterogeneity mix varies among clients.

6.7.6. SCALABILITY ANALYSIS

This scalability analysis explored the performance of FedHIL as the number of clients was increased from 6 to 12 and 18. The 6 Client case had six users with unique smartphone devices, the 12 Client case had twelve devices with each of the six unique smartphone devices used by two out of the 12 users, and the 18 Client case had eighteen devices with each of the six unique smartphone devices used by 3 out of the 18 users. In all scenarios, we set the training device to be MOTO, and the trained model was tested on all participating devices in all available building floorplans.

Figure 59 shows the average localization error per scenario per floorplan. Our findings demonstrate similar localization results across all client devices, with little to no variations across the scenarios. This indicates that the FedHIL framework is scalable, and its performance is stable, even in the presence of scalability of the participating clients. This is an important observation which suggest that the FedHIL framework has great potential for scalability in indoor localization using Wi-Fi RSS fingerprinting.

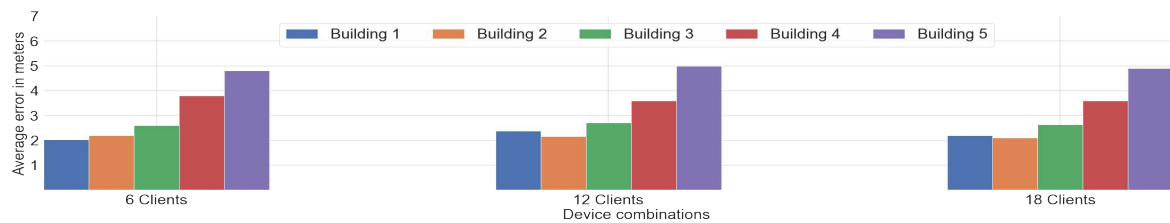


Figure 59: Scalability analysis on different floorplans and varying devices combinations split.

6.8. CONCLUSION

In this chapter, we presented FedHIL, a new FL-based framework that improves indoor localization accuracy while preserving user privacy across diverse environments. FedHIL employs a custom SAE, a lightweight neural network, and a novel aggregation strategy in a distributed learning setting, thereby diversifying the training data and improving the model's accuracy. To address privacy concerns and robustness, FedHIL employs a domain-specific selective weight adjustment approach. We performed extensive analysis with the proposed framework, testing it for heterogeneity, susceptibility to noise, scalability, and skewness, which are common situations faced in real-world distributed learning deployments. Experimental evaluations in real-world settings showed that FedHIL outperformed state-of-the-art FL- and non-FL based frameworks, achieving $1.62\times$ better localization accuracy on average than the best performing FL-based indoor localization framework from prior work. FedHIL thus represents a promising practical framework for simultaneously addressing the many challenges related to heterogeneity, noise-resilience, data privacy, and accuracy in the indoor localization problem domain.

CHAPTER 7

CALLOC: CURRICULUM ADVERSARIAL LEARNING FOR SECURE AND ROBUST INDOOR LOCALIZATION

Indoor localization has become increasingly vital for many applications from tracking assets to delivering personalized services. Yet, achieving pinpoint accuracy remains a challenge due to variations across indoor environments and devices used to assist with localization. Another emerging challenge is adversarial attacks on indoor localization systems that not only threaten service integrity but also reduce localization accuracy. To combat these challenges, we introduce CALLOC, a novel framework designed to resist adversarial attacks and variations across indoor environments and devices that reduce system accuracy and reliability. CALLOC employs a novel adaptive curriculum learning approach with a domain specific lightweight scaled-dot product attention neural network, tailored for adversarial and variation resilience in practical use cases with resource constrained mobile devices. Experimental evaluations demonstrate that CALLOC can achieve improvements of up to 6.03x in mean error and 4.6x in worst-case error against state-of-the-art indoor localization frameworks, across diverse building floorplans, mobile devices, and adversarial attacks scenarios.

7.1. INTRODUCTION

Indoor localization involves the precise determination of the location of devices or individuals within indoor spaces. It has extensive applications across many domains, including asset management, assistive healthcare, navigation within buildings, and the provision of context-aware services [89]. Tech giants such as Apple, Google, and Microsoft are heavily investing in indoor localization research to enhance the accuracy and reliability of indoor location-based services [34].

However, achieving high-location accuracy in real-world indoor scenarios remains an open challenge [89].

Traditional navigation systems, such as Global Positioning System (GPS), heavily rely on satellite signals and clear sky visibility, making them less effective for indoor use. Recognizing this limitation, researchers have redirected their focus towards alternative wireless infrastructures for localization in indoor spaces, including Wi-Fi, Bluetooth, and Zigbee [90]. Among these, Wi-Fi-based localization systems that use received signal strength (RSS) have gained traction due to the widespread availability of Wi-Fi and the ability of modern embedded and IoT devices to capture Wi-Fi RSS [90].

Wi-Fi RSS-based approaches leverage data from Wi-Fi access points (APs) to determine the positions of mobile and IoT devices within indoor environments. In this context, two methodologies have emerged: propagation model-based and fingerprinting model-based localization systems. Propagation model-based systems employ geometric models such as trilateration and triangulation to determine mobile device location using Wi-Fi RSS [91]. Unfortunately, these systems can be susceptible to inaccuracies from environmental factors such as signal variability caused by obstacles, human interference, signal attenuation, multi-path fading, and shadowing [91]. Fingerprinting model-based systems do not rely on geometric models but instead create a database of Wi-Fi signal patterns (“fingerprints”) collected throughout the indoor space. Fingerprinting models are more robust to fading/shadowing/attenuation effects and have demonstrated higher accuracies than propagation model-based methods [90].

Fingerprinting systems consist of two phases: an offline phase for initially constructing the fingerprint database and an online phase for real-time localization. In the online phase, signal patterns may change unpredictably due to factors such as noise in the environment and device

variations (devices capturing dissimilar fingerprints at the same location) [92]. These factors can lead to inaccuracies in location estimations.

To address these limitations, machine learning (ML) techniques can be used to learn from data patterns in the offline phase and adjust predictions in the online phase [36]. By combining the strengths of fingerprinting models with ML, researchers have made significant strides in improving the accuracy and reliability of indoor localization solutions [93].

However, ML-based systems are vulnerable to attacks from adversaries who can exploit vulnerabilities in ML models and subtly alter input data to launch adversarial attacks [94]. In Figure 60, we illustrate the impact of a well-known FGSM adversarial attack [107] on three ML-based indoor localization solutions that use K-Nearest Neighbors (KNN) [17], Gaussian Process Classifier (GPC) [99], and Deep Neural Networks (DNN) [100]. The high loss in accuracy from such attacks can lead to dire consequences, as ML-based indoor localization solutions are often relied upon for decision-making [94], [95].

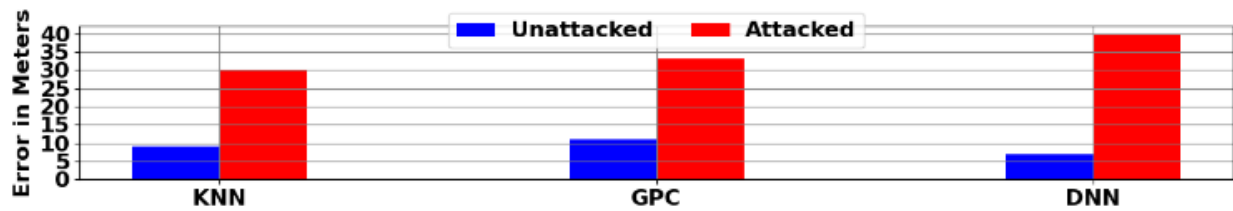


Figure 60: Accuracy reduction (increase in localization error) in three ML-based indoor localization solutions [17], [100] due to adversarial attacks

ML-based RSS fingerprinting indoor localization solutions are also susceptible to other security threats such as AP spoofing, AP signal manipulation, and malicious APs [97]. Mitigating the effect of unpredictable RSS fluctuations due to attacks, as well as environmental and device variations, without compromising localization accuracy remains a formidable challenge. Very few prior

works address security and robustness in ML-based indoor localization solutions. Even fewer address challenges from adversarial attacks.

7.2. CONTRIBUTIONS

In this chapter, we present CALLOC, a novel framework designed to address the challenges posed by environmental noise, device variations, and the often-overlooked aspect of adversarial security during indoor localization. Our novel contributions as part of the CALLOC framework are:

- We develop a novel curriculum learning technique, akin to a knowledgeable teacher guiding a student through progressively complex subjects, to systematically enhance the ML model’s resilience to RSS fluctuations.
- We propose a lightweight domain-specific ML model based on scaled dot product attention neural networks for location prediction on resource constrained devices.
- We perform extensive analysis on real-world data to evaluate CALLOC under different adversarial attacks, mobile devices, and building floorplans, and contrast our performance against the state-of-the-art approaches.

7.3. RELATED WORK

Various classical ML algorithms such as Naïve Bayes [98], KNN [17], and GPC [99] have been investigated to tackle challenges arising from RSS fingerprint variations due to environmental noise from changes over time in density of people, movement of equipment, etc. However, these approaches do not effectively address issues related to device heterogeneity and adversarial attacks. Device heterogeneity refers to situations where two devices capturing Wi-Fi

RSS fingerprints at the same location and time exhibit dissimilar fingerprint patterns, due to disparities in hardware (Wi-Fi chipset) and firmware (noise filtering stack).

More sophisticated ML algorithms based on deep neural networks (DNNs) [100] and convolutional neural networks (CNNs) [101] have emerged in recent years to tackle the complex interplay of environmental noise factors and device heterogeneity. Examples include ANVIL [45] which incorporates a multi-headed attention layer with DNNs, SANGRIA [82] which integrates a domain-specific stacked autoencoder and a categorical gradient-boosted tree classifier, WiDeep [99] which combines a de-noising autoencoder with GPC, and the VITAL framework in [80] which uses vision transformers. These advanced ML approaches are shown to outperform their classical ML counterparts when mitigating environmental factors and device heterogeneity. Nevertheless, these works do not consider the crucial challenges associated with adversarial attacks, which are intended to mislead the ML model and are more challenging to address.

Few efforts have addressed attacks on indoor localization solutions. The frameworks in [103], [88] and [105] address the challenges associated with random AP attacks and data privacy but fail to address challenges from adversarial attacks or environmental and device variations. Adversarial attacks, which involve introducing small perturbations into input data, have been explored in various domains, including computer vision, natural language processing, and edge computing, but not as much for indoor localization [94].

To the best of our knowledge, the only prior works that focus on adversarial attacks during indoor localization are [104] and [106]. Advloc [104] leverages DNNs and incorporates a few adversarial samples into the offline training phase to bolster the ML model's resilience against adversarial attacks in the online phase. The work in [106] also employs DNNs but considers a solution with channel state information (CSI) data instead of RSS, rendering it out of scope for our

study, because most mobile and IoT devices cannot capture Wi-Fi CSI data. These approaches are also entirely based on simulation, and thus exhibit limited resilience against real-world adversarial attacks. They also do not consider real-world challenges posed by environmental variations and device heterogeneity, which further reduce indoor localization accuracy.

Our proposed framework in this chapter overcomes the limitations of the state-of-the-art. We devise a more robust solution for indoor localization, with curriculum learning, which introduces a natural learning order, starting with simpler examples or lessons and progressing toward more intricate ones, thereby enhancing the model's overall learning efficiency progressively. Further, we design a domain-specific attention neural network modified to be lightweight for mobile and IoT device deployment. Our proposed solution is designed to overcome real-world challenges from adversarial attacks and variations across environments and devices.

7.4. ADVERSARIAL ATTACK FORMULATION

Adversarial attacks in indoor localization systems encompass the manipulation or spoofing of wireless signals, typically transmitted by Wi-Fi APs. These attacks can originate from three primary points: the transmitter side (i.e., APs), the receiver side (i.e., mobile devices), or the channel side. In the context of Wi-Fi-based indoor localization, Wi-Fi RSS signals, measured in decibels referenced to one milliwatt (dBm) and typically ranging from -100 dBm (weak signal) to 0 dBm (strong signal), are very susceptible to manipulation, particularly when considering attacks from the channel side, (see example in Figure 61). Channel side attacks offer a distinct advantage due to the open nature of wireless channels, rendering them a preferred choice for adversaries. In contrast, injecting perturbations through the transmitter side (APs) proves more challenging, primarily due to stringent security measures. Meanwhile, attacks on the receiver side necessitate

unauthorized access to personal devices, adding an extra layer of complexity. In our considered attack scenario in this work, we concentrate on channel side attacks within a white-box context. Here, we assume that the attacker possesses comprehensive information about building floorplans, Wi-Fi AP locations, and has access to the ML model's training parameters. This white-box approach is motivated by the attacker's capacity to craft subtle perturbations designed to deceive the ML model with minimal alterations.

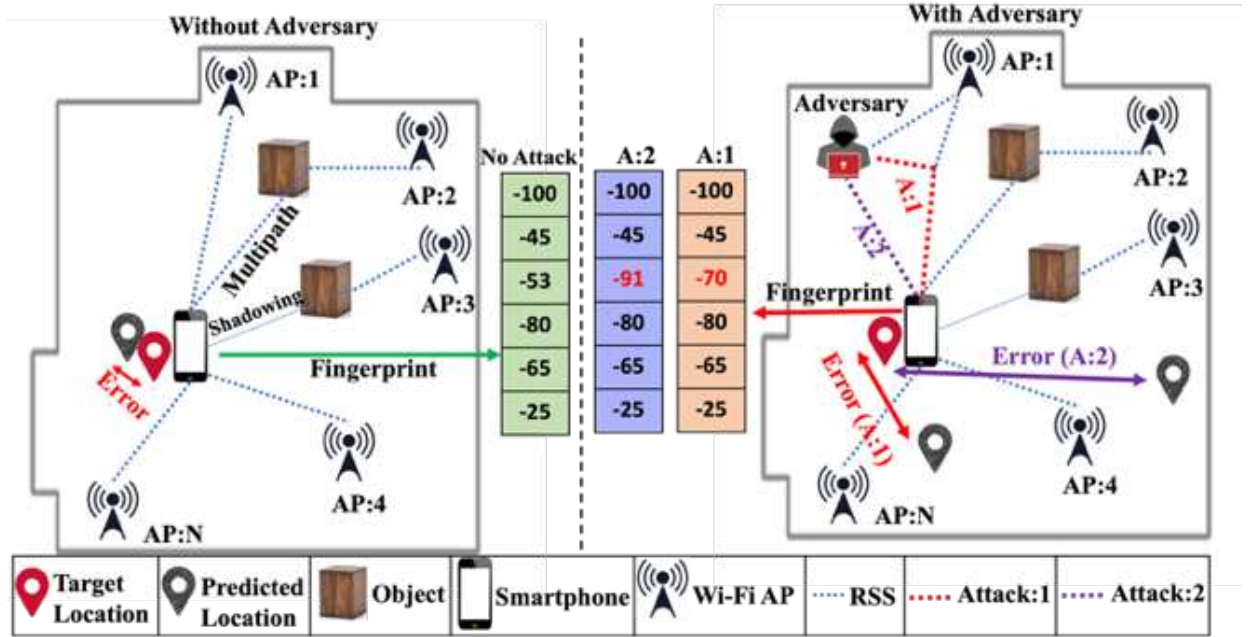


Figure 61: Illustration of indoor localization system with and without adversarial attacks – A:1 (weak attack), A:2 (strong attack)

7.5. ADVERSARIAL ATTACK MECHANISM

In this section, we describe the adversarial attack mechanisms we consider in this work that are capable of compromising the robustness of indoor localization systems. Our analysis centers on man-in-the-middle (MITM) attacks from the channel side. MITM attacks are a well-known class of adversarial techniques wherein an adversary positions themselves between communication flows to manipulate data exchanges. In the context of indoor localization, an adversary can insert

themselves between a mobile device and the targeted Wi-Fi AP used for localization, as shown in Figure 61. By introducing perturbations to the signals exchanged between the device and APs, the attacker can mislead the ML model (deployed on the mobile device) to provide incorrect location estimates. MITM attacks are challenging to detect, especially in large indoor environments.

To the best of our knowledge, this is the first exploration of MITM-based attacks in indoor localization. MITM attacks offer a distinct advantage in the context of a white box scenario, empowering adversaries to craft potent and precisely targeted channel-side attacks. We explore two well-recognized variants: signal manipulation and spoofing attacks:

- **Signal manipulation attacks:** These attacks involve altering Wi-Fi signals to provide misleading information to the indoor localization system. They can distort the RSS data from a Wi-Fi AP, leading to incorrect location estimates. Adversaries may use signal manipulation to misguide the ML model by injecting perturbations into the RSS data, as shown in Figure 61 (A:1).
- **Signal spoofing attacks:** These attacks involve the creation of counterfeit wireless signals designed to closely mimic legitimate APs. In these attacks, adversaries replicate essential characteristics of a target Wi-Fi AP, such as its MAC address and operating channel. By effectively emulating a genuine AP, the spoofing adversary generates their own RSS data, which outwardly resembles that of the legitimate AP. However, the crucial distinction lies in the carefully crafted perturbations introduced into these counterfeit signals. These perturbations are strategically incorporated to mislead the ML model, resulting in inaccurate location estimates, as shown in Figure 61 (A:2). This method contrasts with signal manipulation attacks,

where genuine RSS data is directly tampered with, whereas signal spoofing attacks involve the fabrication of signals that closely resemble legitimate ones.

7.5.1. WHITE-BOX ADVERSARIAL ATTACK FORMULATION

In the context of indoor localization security, as discussed above, the main strategy employed in various attack scenarios involves altering RSS data with carefully planned modifications to deceive the ML model. To recreate these scenarios in a real-world white-box environment, we employ three popular methods for crafting white-box adversarial attacks. These methods include FGSM [107], PGD [108], and MIM [109]. Each of these techniques serves a unique purpose in aiding our understanding of adversarial attacks on indoor localization systems. These techniques are described below:

- **Fast gradient sign method (FGSM):** FGSM is a one-step, non-iterative attack method that calculates the adversarial perturbation based on the gradient of the ML model loss function. It then applies this perturbation directly to the input data. It is employed in indoor localization to perturb the RSS data from a targeted Wi-Fi AP. The attack focuses on manipulating RSS data at a single step by adding small but deliberate changes. The FGSM attack can be mathematically represented as follows:

$$X_{Adv} = X + \epsilon * \text{sign}(\nabla J(X, Y)) \quad (29)$$

where X_{Adv} is the adversarial example (perturbed RSS data), X is the original RSS data, ϵ is the magnitude of the perturbation, $\nabla J(X, Y)$ are the gradients of the loss function, and Y is the true label (actual location).

- **Projected gradient descent (PGD):** PGD is an iterative adversarial attack method that refines the perturbation in multiple steps. In each step, it calculates the gradients of the ML model loss function with respect to the input data and updates the perturbation. This iterative process aims to find a stronger perturbation for deceiving the model. Unlike FGSM, which operates in a single step, PGD iteratively enhances the perturbation. The PGD attack can be mathematically represented as follows:

$$X_{Adv} = clip\{X + \alpha * (\nabla J(X, Y), \epsilon)\} \quad (30)$$

where X (original RSS data) is adjusted iteratively. The perturbations are controlled by α (alpha) and capped (*clip function*) at a perturbation magnitude ϵ .

- **Momentum iterative method (MIM):** MIM is another iterative adversarial attack approach designed to refine perturbations across multiple steps. In each iteration, it computes the gradients of the ML model loss function relative to the input data and updates the perturbation, gradually strengthening it to deceive the model effectively. MIM distinguishes itself from PGD by introducing momentum into the optimization process, which allows it to converge more efficiently and find effective perturbations, making it particularly potent for crafting adversarial examples. The MIM attack can be mathematically represented by Equation 30 from PGD.

7.5.2. ADVERSARIAL DATA GENERATION

To evaluate the indoor localization systems' resilience against MITM attacks and their variants, we harness FGSM, PGD, and MIM to craft adversarial data. These techniques allow us to generate MITM-like attacks and systematically assess the robustness of our system. One crucial parameter in crafting adversarial examples is ϵ , representing the attack strength. Varying ϵ from 0.1 to 0.5 allows us to examine the impact of different attack strengths on the system. As we increase ϵ , the attack strength intensifies, causing perturbations in RSS data to become more pronounced, as shown in Figure 61 (A:2 has higher ϵ than A:1). By encompassing this range of ϵ values, we are able to explore the system's response to a spectrum of adversarial intensities.

Additionally, we consider the parameter $\varnothing$, which denotes the number of targeted Wi-Fi APs chosen for attack. The example in Figure 61 shows the case of $\varnothing = 1$. This selection of specific APs is essential as it mirrors the attacker's choice in the real world. Crafting adversarial data for this subset of APs allows us to evaluate the model's robustness in scenarios where certain APs are compromised. By assessing the system's performance against MITM attacks with varying ϵ and $\varnothing$, we can gain insights into a framework's capacity to withstand adversarial threats across different conditions.

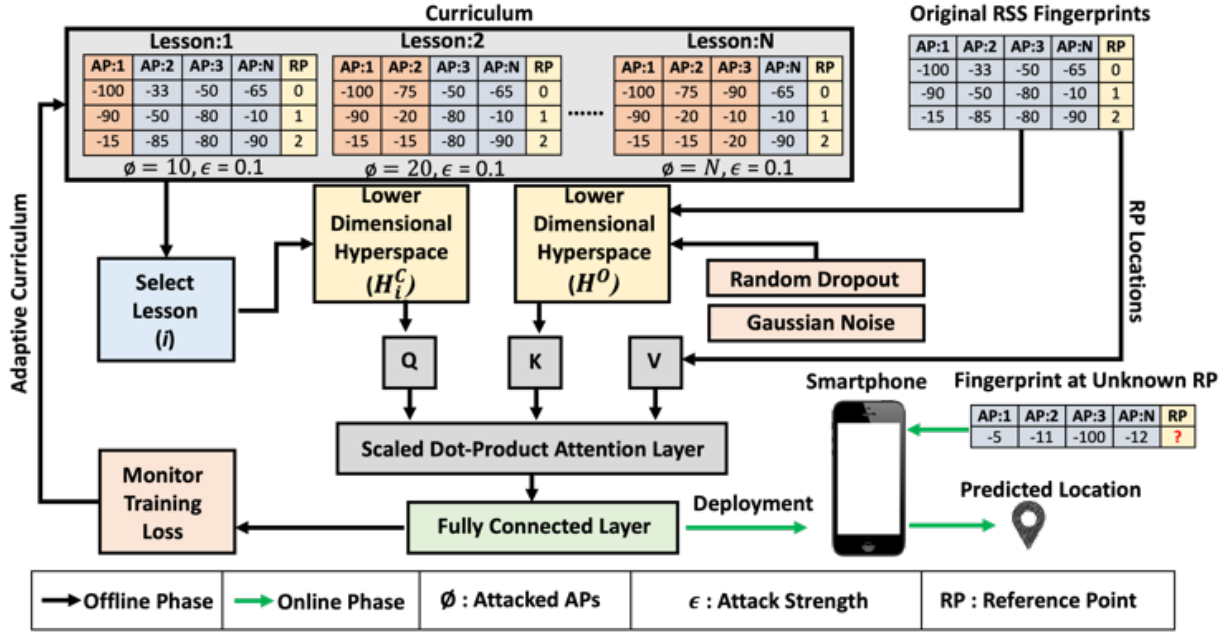


Figure 62: Overview of the CALLOC framework: offline and online phases

7.6. CALLOC FRAMEWORK

The CALLOC framework consists of two phases: an offline and an online phase, as shown in Figure 62. In the offline phase, we begin by collecting RSS fingerprints from various reference points (RPs) in a building using a single device (see Section 7.7). Initially, this data collection occurs without any adversarial interference ($\phi = 0, \epsilon = 0$). To enhance model robustness against adversarial threats, we employ a novel curriculum that systematically introduces attacks on different APs (ϕ) (see Section 7.6.1). This curriculum has a dual purpose: it guides the model's learning trajectory and progressively challenges it with escalating adversarial scenarios by targeting different subsets of APs (increasing ϕ).

Once the lesson is selected from the curriculum, we utilize domain-specific neural networks (see Section 7.6.2) to transform both the lesson and the original attack-free data into lower-dimensional hyperspaces: H_i^C (curriculum data) and H^O (original data), as shown in Figure 62,

where i is the lesson number. H_i^C captures essential curriculum features, while H^O captures original data features along with random dropouts and Gaussian noise, to simulate real-world environmental and device variations. This augmentation enhances the model’s adaptability to practical scenarios.

These hyperspaces, along with RP locations, serve as inputs for the attention layer within the model, requiring Q (query), K (key), and V (value) inputs (see Section 7.6.3). H_i^C is assigned to Q , H^O to K , and RP locations to V . The attention mechanism enables the model to focus on relevant information within the hyperspaces and RP locations, allowing it to capture crucial patterns that contributes towards accurate indoor localization predictions. The attention output is then passed to a fully connected layer responsible for predicting RP locations for unknown fingerprints collected during the online phase, as shown in Figure 62. Throughout the training process, we closely monitor the loss function of the final fully connected layer (see Section 7.6.4). This monitoring continues until the final lesson is completed.

In summary, *CALLOC*’s approach involves a novel curriculum-based learning and domain-specific hyperspace transformations with attention, to simultaneously build resilience to adversarial threats, environmental variations, and device heterogeneity, while being lightweight for deployment on mobile and IoT devices. The following subsections describe the framework in more detail.

7.6.1. CURRICULUM SELECTION

Our curriculum comprises of 10 distinct lessons. The curriculum initiation begins with a baseline lesson featuring 0% attacked APs ($\varnothing = 0$) and 100% original data. Subsequent lessons contain higher $\varnothing$ and lower number of original data. For example, the second lesson contains $\varnothing =$

10 (10% attacked APs) with $\epsilon = 0.1$. This progression culminates in the toughest scenario at lesson 10, with $\varnothing = 100$ and $\epsilon = 0.1$. By maintaining a fixed and small ϵ value during training, we simulate subtle perturbations similar to those an adversary might introduce in the online phase. As we transform this perturbed data into a lower-dimensional hyperspace (see Section 7.6.2), we capture nuanced changes in essential features within the curriculum data (influenced by ϵ) compared to the original data. This guides the model to learn how perturbations can affect the RSS inputs. Consequently, CALLOC becomes resilient against similar perturbation patterns, even when the attack magnitude (ϵ) varies, ultimately enhancing its robustness during testing.

Adversarial data is generated using the FGSM technique, ensuring that the model learns to defend against adversarial attacks. Note that our approach does not require exposure to the many other types of adversarial data (e.g., from PGD and MIM) during training, while still showing robustness to those attacks in the online phase (see Section 7.7). The selection of lessons follows a systematic approach, commencing with lesson 1, where the model learns to handle original data effectively. After completing each training phase, the model proceeds to the next lesson in a progressive manner until it has undergone training across all the lessons.

7.6.2. LOWER-DIMENSIONAL HYPERSPACE

Once a lesson is selected for training, the next step involves mapping the lesson data into a lower-dimensional hyperspace, a crucial step in our training approach. This mapping aims to retain essential information while reducing the dimensionality of the input data, making it more computationally efficient and reducing the risk of overfitting. We employ two specialized embedding neural networks for this purpose: one dedicated to the curriculum data and another to the original data. These networks are designed with a reduced number of neurons compared to the

input features. The primary motivation behind this dimensionality reduction is twofold. First, it enhances the model's ability to generalize effectively, by focusing on essential features and patterns in the data. Second, reducing dimensionality serves as a means of noise reduction. Noise in the data can originate from environmental factors or device variations. Additionally, these hyperspaces offer computational efficiency, a crucial consideration, especially for mobile and IoT device.

The embedding network for the original data incorporates dropout and Gaussian noise layers. Dropout randomly removes some neuron outputs during training, preventing the model from relying too heavily on certain input features, thus preventing overfitting. The Gaussian noise layers add variability to the training process, further enhancing the model's robustness across different conditions.

7.6.3. SCALED DOT-PRODUCT ATTENTION NEURAL NETWORK

The hyperspaces generated from the curriculum data and the original data represent lower-dimensional representations of the input data, respectively. To measure similarities between these hyperspaces, we employ a scaled dot-product attention neural network. This attention mechanism takes three inputs: Q (query), K (key), and V (value). By computing similarities between elements in Q (H_i^C) and K (H^O), the attention layer enables the network to prioritize relevant features in the hyperspaces and their corresponding RP locations. This mathematical process can be expressed as:

$$Attention(Q, K, V) = Softmax\left(Q K^T / \sqrt{d_k}\right) V \quad (31)$$

In this equation, Q represents H_i^C , K represents H^O , and V signifies RP locations. The dot product between Q and K , followed by scaling ($\sqrt{d_k}$), generates attention scores. These scores, after applying the *Softmax* function, become the attention weights that capture the model's focus on relevant information. Higher attention weights suggest that the model is giving more importance to certain elements in the hyperspaces. The attention output, consisting of weighted information, is then channeled to a fully connected layer. This layer is responsible for classifying and predicting RP locations for unknown fingerprints in the online phase.

7.6.4. ADAPTIVE CURRICULUM

Throughout the training process, we carefully monitor the loss function of the final fully connected layer. This monitoring process is vital as any observed increase in loss during training can be an early indicator of a potential divergence in the training process. Divergence may indicate that the model is struggling to adapt to specific data patterns, which can be influenced by σ in the lessons. In response to this, CALLOC employs an early stopping mechanism to safeguard optimal performance. When such an increase in loss is detected, the model reverts to its best-performing weights, ensuring that it maintains peak performance.

Moreover, CALLOC's curriculum is not static; it is a responsive component that fine-tunes the model's training approach. After reverting to the best weights, the curriculum is carefully adjusted by reducing σ by steps of two in the lesson data. This adjustment allows for the management of data complexity and further improves the model's adaptability. The curriculum adapts alongside the model, ensuring that the training process is tailored to the evolving needs of the model. Once

the training process successfully reduces loss, indicating improved adaptation, the model advances to the next lesson. This dynamic procedure continues iteratively until the final lesson is completed, resulting in a model well-prepared to handle uncertainties in real-world scenarios.

7.7. EXPERIMENTAL SETUP

In our experiments, we evaluate CALLOC's performance in real-world scenarios across various adversarial attacks. We also compare CALLOC against state-of-the-art frameworks, including AdvLoc [103], SANGRIA [82], ANVIL [45], and WiDeep [100]. CALLOC exclusively trains on adversarial samples generated via FGSM, maintaining a small constant ϵ value of 0.1 throughout the curriculum while varying σ . Furthermore, we conduct an extensive analysis of CALLOC's resilience across diverse mobile devices and building floorplans. The details of the heterogeneous mobile devices selected for evaluation are presented in Table 12.

Table 12: Smartphones details used to evaluate the CALLOC framework.

Manufacturer	Model	Acronym
BLU	Vivo 8	BLU
HTC	U11	HTC
Samsung	Galaxy S7	S7
LG	V20	LG
Motorola	Z2	MOTO
Oneplus	3	OP3

The building floorplans considered in our experiments encompass variations in path length, visible APs, and environmental noise characteristics (see Table 13). Data collection involves the devices selected, with an allocation of 5 fingerprints per RP per building for training and reserving 1 fingerprint per RP per device per building for testing purposes. The OP3 device serves as our

designated device to capture training data. The RPs maintain a physical granularity of 1 meter. We plan to open-source the dataset from this study, to benefit the indoor localization community [96].

Table 13: Building floorplan details used to evaluate the CALLOC framework.

Building Number	Visible APs	Path Length	Characteristics
Building 1	156	64 meters	Wood and Concrete
Building 2	125	62 meters	Heavy Metallic Equipments
Building 3	78	88 meters	Wood, Concrete, Metal
Building 4	112	68 meters	Wood, Concrete, Metal
Building 5	218	60 meters	Wide Spaces, Wood, Metal

CALLOC utilizes domain-specific embedding neural networks, each composed of 128 neurons, utilizing mean square error as the loss function for both hyperspaces. In the H^O network, we empirically determine and set dropout rate to 0.2 and Gaussian noise to 0.32. The final fully connected layer is designed to classify the RP classes, resulting in a total of 65,239 trainable parameters. These parameters are distributed as follows: 42,496 trainable parameters in both the embedding layers, 18,961 in the attention layer, and 3,782 in the final fully connected layer, achieving a compact model size of 254.84 kB (kilobytes), for efficient deployment.

7.7.1. CALLOC EVALUATIONS: DEVICES, FLOORPLANS AND ATTACKS

In this section, we assess CALLOC's performance across smartphones and buildings under different adversarial attack methods. Figure 63 presents heatmaps corresponding to changes in indoor localization accuracy under FGSM, PGD, and MIM attacks. The heatmaps display the mean localization error in meters. The experiments are conducted over varying ϵ (ranging from 0.1 to 0.5) and σ (ranging from 10 to 100) values, to represent diverse real-world adversarial attack scenarios. The tests across multiple buildings are meant to capture a variety of environmental noise

scenarios. Lastly, the model is trained on the OP3 device and tested on all devices, to capture scenarios of noise due to device heterogeneity.

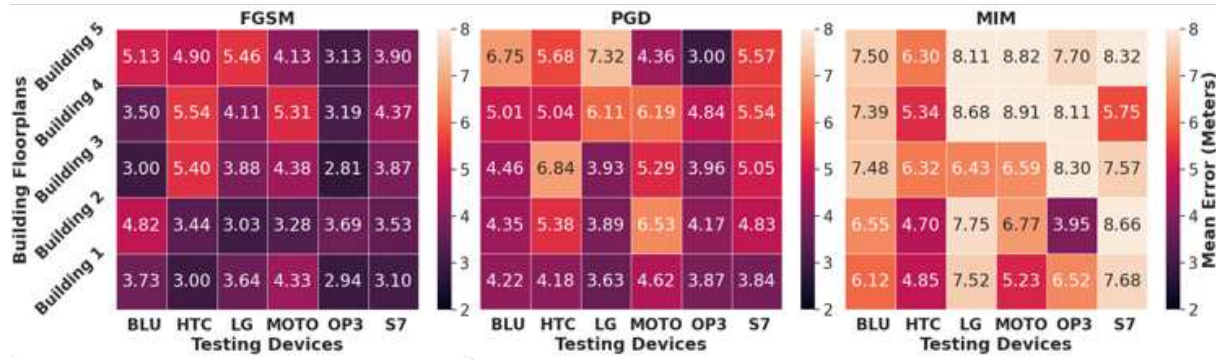


Figure 63: Localization errors across different devices, buildings, and attacks

In general, as expected, localization accuracy for CALLOC reduces in the presence of attacks, environmental variations, and device heterogeneity. However, CALLOC is able to limit the accuracy degradation in the presence of variations (as will also become apparent in later results that contrast it with the performance of other frameworks). The heatmaps show that CALLOC demonstrates resilience to device heterogeneity, as it maintains consistent performance regardless of the testing devices (i.e., low errors across a row in the heatmaps). Specific devices do exhibit higher variations in error rates e.g., FGSM-Building1-OP3 and FGSM-Building1-MOTO. This behavior can arise due to extreme heterogeneity in wireless chipsets and software stacks across devices, that can cause the captured RSS data from these devices to vary significantly. Additionally, certain building floorplans may exhibit higher errors compared to others, e.g., Building 1 and Building 5. This behavior is due to different salient features in the building floorplans contributing to added noises (e.g., greater dynamic density of people, movement of equipment). Across the three attacks, FGSM has the least error, while stronger iterative attacks

such as PGD and MIM result in slightly higher errors compared to FGSM. Nonetheless, CALLOC, equipped with its specialized curriculum and hyperspace-attention model, effectively reduces the impact of device heterogeneity and environmental noise, as well as unpredictable adversarial attacks, by limiting the increase in indoor localization error.

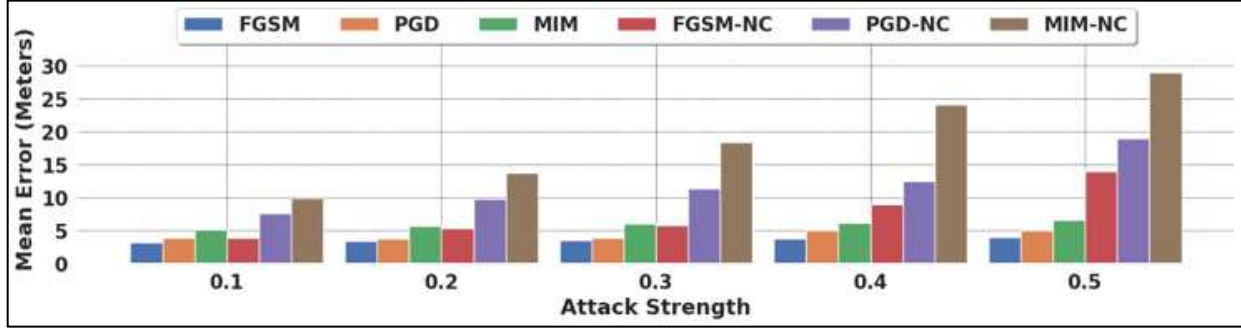


Figure 64: Impact of curriculum learning on different attacks and ϵ

7.7.2. EVALUATING IMPACT OF CURRICULUM LEARNING WITH ϵ

Next, we assess the impact of the proposed curriculum learning technique across varying attacks and ϵ values. Figure 64 presents a plot depicting the responses of different attacks when subjected to different ϵ values. Each bar represents the mean response across all testing devices, building floorplans, and σ values ranging from 10 to 100. We also compare these results with the performance of the CALLOC framework when curriculum learning is not applied, denoted as 'NC' (No Curriculum). The trends from the results reveal that the incorporation of curriculum learning bolsters CALLOC's resilience against varying ϵ values and diverse attack methods, outperforming the 'NC' approach. Notably, in the absence of curriculum learning, the model exhibits pronounced susceptibility to different attacks, particularly in scenarios with higher ϵ values. These findings highlight the efficacy of curriculum learning to withstand adversarial attacks and variations in ϵ values.

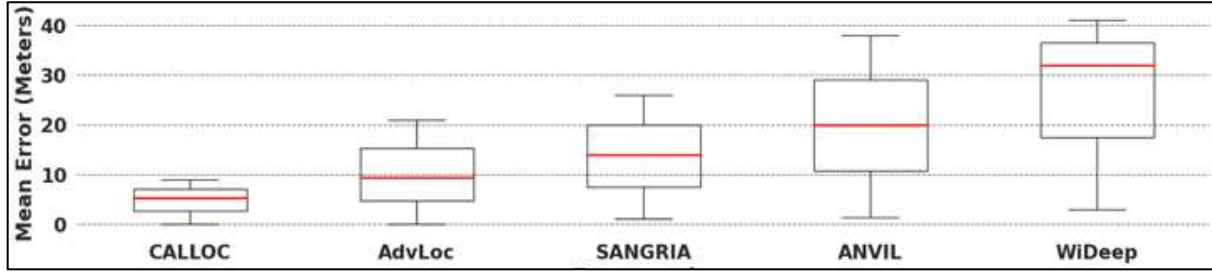


Figure 65: Comparing CALLOC against state-of-the-art frameworks.

7.7.3. COMPARING CALLOC AGAINST STATE-OF-THE-ART

We evaluate CALLOC's performance against state-of-the-art frameworks across different devices, buildings, ϵ (0.1 to 0.5), and σ (1 to 100), as shown in Figure 65. It can be observed that CALLOC consistently outperforms its competitors, showing the lowest maximum (worst-case), and mean errors. Notably, CALLOC surpasses AdvLoc by $1.77\times$ (mean) and $2.35\times$ (worst-case), due to its novel curriculum learning strategy and domain-specific hyperspace-attention model. It also outperforms SANGRIA, which excels in augmentation to noise but lags in adversarial robustness, by $2.64\times$ (mean) and $2.92\times$ (worst-case). CALLOC outperforms ANVIL, which provides exceptional heterogeneity and noise resilience due to its strategic multi-headed attention network, but lags in mitigating adversarial attacks, by $3.77\times$ (mean) and $4.26\times$ (worst-case). CALLOC outperforms WiDeep by a remarkable $6.03\times$ (mean) and $4.6\times$ (worst-case). This is due to the incorporation of GPC in WiDeep, which is extremely sensitive to noise. These results highlight CALLOC's exceptional performance in real-world scenarios.

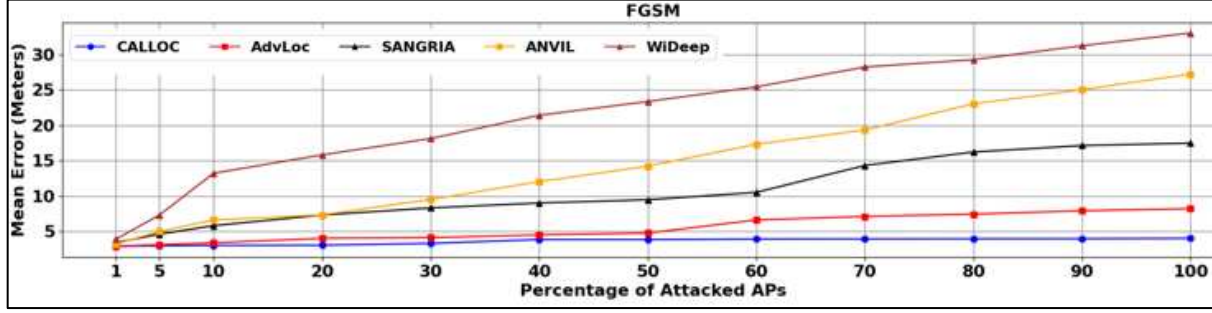


Figure 66: Effect of varying attacked APs ($\varnothing$) on localization error

7.7.4. EVALUATING IMPACT OF VARYING $\varnothing$

We further evaluate CALLOC's response to variations in the number of attacked APs ($\varnothing$) using different attack methods, covering $\varnothing$ values ranging from 1 to 100, and compare the results with state-of-the-art frameworks. Figure 66 illustrates CALLOC's response for different $\varnothing$ values alongside the performance of other frameworks. It is evident that even small increments in $\varnothing$ can significantly reduce localization accuracy. CALLOC, in response to the FGSM attack, exhibits relatively stable errors as $\varnothing$ increases, unlike other frameworks. AdvLoc, another framework that also incorporates a subset of FGSM samples into its training, was surpassed by CALLOC across FGSM data and more potent attacks like PGD and MIM. AdvLoc shows a slightly high error than CALLOC, with the error increasing starting at $\varnothing = 60$. ANVIL, SANGRIA, and WiDeep show higher errors for both low and high values of $\varnothing$. These trends hold for PGD and MIM attacks as well (result plots omitted for brevity). In summary, CALLOC's resilience to adversarial attacks set it apart from other frameworks from prior work.

7.8. CONCLUSION

Our proposed CALLOC framework showcases resilience against adversarial attacks, variations across devices, and noise across indoor environments. It surpasses achievable accuracy over state-of-the-art localization frameworks by a substantial margin, showing improvements of up to $6.03\times$ in mean errors and $4.6\times$ in worst-case errors. Through rigorous evaluation, we observed that CALLOC consistently maintains lower localization errors across devices, buildings, and adversarial attack vectors. The implementation of curriculum learning significantly enhances CALLOC's robustness, while being lightweight for mobile and IoT device deployment.

CHAPTER 8

SENTINEL: SECURING INDOOR LOCALIZATION AGAINST ADVERSARIAL ATTACKS WITH CAPSULE NEURAL NETWORKS

With the increasing demand for edge device powered location-based services in indoor environments, Wi-Fi received signal strength (RSS) fingerprinting has become popular, given the unavailability of GPS indoors. However, achieving robust and efficient indoor localization faces several challenges, due to RSS fluctuations from dynamic changes in indoor environments and heterogeneity of edge devices, leading to diminished localization accuracy. While advances in machine learning (ML) have shown promise in mitigating these phenomena, it remains an open problem. Additionally, emerging threats from adversarial attacks on ML-enhanced indoor localization systems, especially those introduced by malicious or rogue access points (APs), can deceive ML models to further increase localization errors. To address these challenges, we present SENTINEL, a novel embedded ML framework utilizing modified capsule neural networks to bolster the resilience of indoor localization solutions against adversarial attacks, device heterogeneity, and dynamic RSS fluctuations. We also introduce RSSRogueLoc, a novel dataset capturing the effects of rogue APs from several real-world indoor environments. Experimental evaluations demonstrate that SENTINEL achieves significant improvements, with up to $3.5\times$ reduction in mean error and $3.4\times$ reduction in worst-case error compared to state-of-the-art frameworks using simulated adversarial attacks. SENTINEL also achieves improvements of up to $2.8\times$ in mean error and $2.7\times$ in worst-case error compared to state-of-the-art frameworks when evaluated with the real-world RSSRogueLoc dataset.

8.1. INTRODUCTION

In recent years, indoor localization has gained attention for its versatile applications across several industries, such as healthcare, asset tracking, smart homes, location-based advertising, and much more [89]. The ability to pinpoint the exact location of edge devices within indoor settings has the potential to revolutionize these industries and elevate user experiences significantly. Hence, technology giants such as Apple, Google, Meta, and Microsoft are making substantial investments in indoor localization research to improve the accuracy and reliability of indoor location-based services [34]. However, achieving high-precision indoor localization remains a formidable challenge due to the inherent complexities and dynamic nature of indoor environments.

Traditional navigation systems, such as the global positioning system (GPS), have found widespread adoption in popular tools such as Google Maps, Apple Maps, and Waze, mainly owing to their commendable localization accuracies in outdoor settings. However, the dependence of GPS on satellite signals and clear sky visibility poses a significant limitation, rendering this approach ineffective for indoor use [110]. In response to this challenge, researchers have shifted their attention to alternate wireless infrastructures that could be a better fit for localization across indoor spaces, such as Wi-Fi, Bluetooth, and Zigbee. Among these alternatives, Wi-Fi-based localization systems utilizing received signal strength (RSS) have gained significant traction [89], [111]. This surge in popularity for this solution is attributed to the ubiquitous availability of Wi-Fi in indoor spaces and the capability of modern edge devices to capture Wi-Fi RSS, making it a viable option for indoor localization [111].

Wi-Fi RSS is obtained by measuring the signal strength of nearby Wi-Fi routers or access points (APs) via edge devices. This captured RSS data can be used to estimate the current indoor

location of an edge device. As the edge device moves, it periodically captures new RSS measurements, reflecting the edge device's mobility. Leveraging this changing RSS data, many techniques have been proposed for accurate indoor localization, with geometric model-based [112] and fingerprinting model-based [111], [37] approaches emerging prominently. Geometric models utilize propagation methods such as trilateration [113] and triangulation [114] to pinpoint an edge device's location. However, these solutions are prone to inaccuracies as they are particularly sensitive to RSS fluctuations caused by dynamic changes and complexities within indoor environments. On the other hand, fingerprinting model-based systems eschew propagation methods by creating a database of Wi-Fi RSS patterns ("fingerprints") of visible Wi-Fi APs collected throughout the indoor space to estimate location. Fingerprinting models have been shown to exhibit greater resilience to RSS fluctuations, demonstrating higher accuracies than geometric methods [111], [115].

Fingerprinting-based localization solutions comprise of two distinct phases: an offline phase and an online phase. During the offline phase, Wi-Fi RSS fingerprints are systematically captured across multiple reference points (RPs) within a building floorplan. Typically, multiple fingerprints are recorded per RP to accommodate data variability that can arise due to RSS fluctuations in the online phase. These fingerprints are then often utilized to train a machine learning (ML) model, enabling it to capture underlying patterns and features within the collected RSS fingerprints [36]. Once trained, this ML model is deployed on the edge device, making it available in the online phase for real-time indoor location predictions.

In the online phase, the RSS fingerprints may exhibit fluctuations due to diverse factors in the indoor environments. These factors include signal attenuation, reflections from objects, human interference, and multipath fading, which can introduce fluctuations in the collected RSS

fingerprints [116]. Furthermore, edge device heterogeneity exacerbates this issue. Even among edge devices utilizing the same Wi-Fi chipset (from the same manufacturer), differences in hardware, software, antenna configurations, and firmware settings can introduce fluctuations in RSS fingerprints [116]. As a result, training an ML model can be challenging as heterogeneous and noisy RSS can result in poor generalization and result in inaccurate location predictions. Priors works have shown up to a 41% reduction in location accuracy due to these factors [117]. Additionally, the often-overlooked factor of adversarial attacks can not only perturb the RSS fingerprints (thereby introducing stronger fluctuations) but also compromise the accuracy and effectiveness of localization with the edge device, emphasizing the need for more robust and secure localization systems.

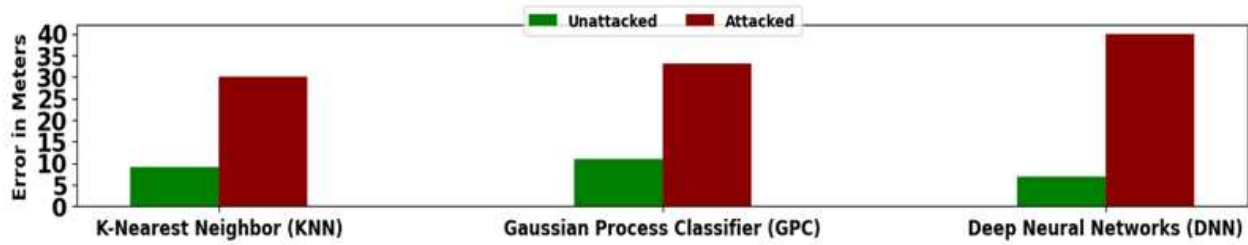


Figure 67: Impact of rogue APs on three popular ML-based indoor localization solutions [119], [120], [121] from prior work.

Adversarial attacks can mislead popular ML models, including state-of-the-art deep learning (DL) algorithms that have been shown to be vulnerable to adversarial examples. The authors of [107] verified the discovery by misleading the popular GoogLeNet [118] model with adversarial examples. Similarly, ML based indoor localization systems also face the threat of adversarial attacks. The presence of malicious (or rogue) APs in the building floorplan can be used to create adversarial attacks by mimicking a legitimate AP and broadcasting erroneous RSS values. In

Figure 67, we illustrate the detrimental impact of the presence of rogue APs on three popular ML-based indoor localization solutions based on K-Nearest Neighbors (KNN) [119], Gaussian Process Classifier (GPC) [120], and Deep Neural Networks (DNN) [121]. This experiment was conducted on an indoor path in a building measuring 55 meters in length containing 55 RPs (1 RP per meter), with up to 203 visible APs (per RP). The experiment incorporated the popular fast gradient sign method (FGSM) [129] technique to simulate the presence of rogue APs, resulting in significantly increased indoor localization errors, with average error increases of $3.33\times$ for KNN, $3.0\times$ for GPC, and $5.71\times$ for DNN, highlighting the negative impact of the rogue APs on localization accuracy.

8.2. CONTRIBUTIONS

To tackle the challenges posed by RSS fluctuations in dynamic indoor environments, edge device heterogeneity, and rogue AP attacks, in this chapter we introduce SENTINEL, a novel embedded ML framework that employs modified capsule neural networks tailored specifically for indoor localization and rogue AP resilience, offering a more practical, secure, and real-time solution for indoor localization. The major contributions of our SENTINEL framework are:

- We design a novel modified capsule neural network specifically for the RSS fluctuation challenges in indoor localization, tailored to 1) overcome the spatial invariance problem in prior DL-based indoor localization efforts and 2) enable lightweight deployment on edge devices.
- We study the effects of rogue AP attacks and propose an adversarial training setup together with the modified capsule neural network for resilience against adversarial (rogue) AP attacks for the first time in indoor localization.

- We introduce a new Wi-Fi RSS fingerprint dataset called RSSRogueLoc [59] that captures AP attacks from rogue APs in real-world indoor environments for the first time.
- We conduct a performance comparison with SENTINEL against state-of-the-art indoor localization solutions, to highlight its effectiveness in the presence of diverse adversarial attacks, edge device heterogeneity, and RSS fluctuations across diverse indoor building paths.

8.3. RELATED WORK

Wi-Fi fingerprinting-based indoor localization has gained significant recognition, evident in competitions hosted by industry giants like Microsoft and NIST [34]. Several classical ML-based solutions, such as ones based on the KNN [119] and GPC [120] algorithms have showcased their potential in addressing RSS fluctuations arising from dynamic effects in indoor environments. These fluctuations encompass various factors, including human interference, obstacles, movement of furniture or equipment, variable population density, signal interference, reflections by objects, and shadowing effects [123], [88], [134].

Despite the demonstrated promise of these ML solutions, they often face challenges in maintaining robustness against fluctuations introduced by edge device heterogeneity. The heterogeneity issue arises from differences in Wi-Fi chipsets and noise filtering software employed by different manufacturers of edge devices. As these chipsets and software stacks are crucial for extracting RSS fingerprints [116], [123], the heterogeneity within them introduces additional complexities for traditional ML-based indoor localization systems.

In response to these challenges, researchers have explored the use of more powerful DL algorithms for indoor localization, including DNNLOC [121], MLPLOC [122], LC-DNN [123], CNNLOC [28], SANGRIA [82], ANVIL [45], and TIPS [39]. DNNLOC [121], MLPLOC [122],

and LC-DNN [123] employ DNNs along with improved RSS pre-processing methods to enhance feature correlation in the RSS fingerprints. CNNLOC [28] proposes a modified convolutional neural network (CNN), to improve on these efforts by enhancing the model's ability to capture relevant features in the RSS fingerprints. SANGRIA [82] employs DNN based autoencoders while ANVIL [45], [39] utilizes attention neural networks, to improve focus on critical input features. TIPS [39] leverages transformer-based encoding of RSS fingerprints for improved resilience against fluctuations introduced by dynamic indoor environments and device heterogeneity. However, these approaches are still significantly impacted by more complex heterogeneity effects in emerging devices and are also susceptible to adversarial attacks, due to the spatial invariance problem in DL algorithms.

Most DL algorithms, particularly CNNs, suffer from the spatial invariance problem where the DL algorithm has a propensity to focus solely on the presence of features in the data while neglecting the precise relative positions of the features [125]. Alterations in the position of each feature can lead to mispredictions by the DL model. This limitation is illustrated in Figure 68, where the VGGFace algorithm [126], a CNN-based model, struggles to differentiate between the two faces. In the Figure on the left, a normal human face is depicted, while the Figure on the right presents an abnormal face with jumbled feature positions. The model assigns the same output classification probability to both cases. The concern regarding feature positions is particularly relevant in the context of RSS fingerprints for indoor localization, where positions of certain features represent crucial information and can be specific to a particular RP. When an edge device moves to a different RP, the positions of these features may undergo changes based on the characteristics of the new RP location. Thus, it is imperative to account for the dynamic nature of feature positions when designing practical indoor localization solutions.

Human Face : 95.88% Human Face : 95.88%

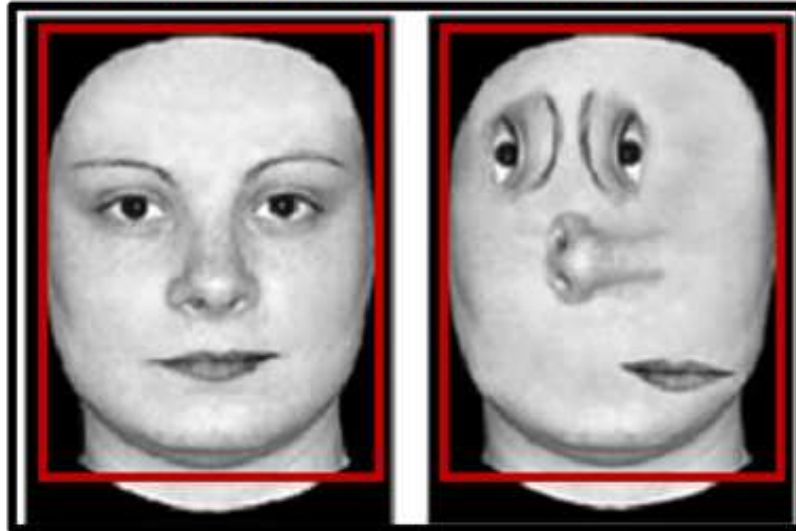


Figure 68: Spatial invariance problem in deep learning algorithms. Both cases are classified as valid human faces by a CNN model.

To address this limitation and enhance feature extraction, researchers have embraced more recent DL algorithms, such as vision transformers (VITAL) [80], [136] and capsule neural networks (EDGELOC) [127] for indoor localization. VITAL [80], uses vision transformers, introduces positional encoding for each feature, aiming to overcome the spatial invariance limitations posed by CNNs. Similarly, EDGELOC [127] uses a simple capsule neural network derived from [133], treating each captured feature as a vector, considering both magnitude and direction of features. These frameworks show the potential to greatly mitigate the effects of dynamic environments and heterogeneity for indoor localization. However, the introduction of adversarial attacks especially arising from rogue APs can not only jumble the feature positions but also introduce new malicious features in the data. Such attacks can easily mislead state-of-the-art localization frameworks and compromise user security.

Adversarial training has emerged as a potential solution to address the challenges from adversarial attacks in ML [128]. Popular solutions typically incorporate a subset of adversarial samples along with the training data to allow robustness in the presence of adversarial attacks during inference. Adversarial samples are generated using several popular adversarial methods out of which the fast gradient signed method (FGSM) [129] has been widely employed to simulate the effects of adversarial attacks, owing to its simplicity. ADVLOC [130] and CALLOC [87] are two recent solutions that incorporate adversarial training, aiming to address the effects of adversarial attacks in indoor localization. Both ADVLOC [130] and CALLOC [87] integrate FGSM samples during training for adversarial resilience. CALLOC additionally employs curriculum learning along with attention neural networks to enhance feature extraction between the original and adversarial samples, to improve overall robustness. Nevertheless, both solutions fall short of addressing the multitude of challenges associated with dynamic environments, heterogeneity, and adversarial attacks concurrently. Additionally, these solutions heavily rely on simulated data for measuring the efficacy of the model's performance against adversarial attacks in the online phase. Their performance in real-world adversarial scenarios has not yet been carefully studied.

After carefully studying the simultaneous challenges of dynamic environments, edge device heterogeneity, adversarial attacks, and lack of real-world adversarial attack data to measure the effectiveness of adversarial resilience in indoor localization, in this work we propose SENTINEL, a novel embedded ML framework that goes beyond state-of-the-art DL solutions to better address the spatial invariance problem and improve robustness using an enhanced capsule neural network with techniques that more comprehensively improve resilience to real-world indoor localization challenges. Another important contribution of our work is the design of a newly curated RSS fingerprint dataset called RSSRogueLoc [59] that captures the presence of rogue APs within indoor

building paths, to analyze the impact of adversarial attacks on indoor localization frameworks in real-world environments, for the first time.

8.4. ADVERSARIAL ATTACKS IN INDOOR LOCALIZATION

Adversarial attacks involve deliberately perturbing input data to deceive an underlying ML model [129]. This perturbation typically consists of adding noise to individual data values (datapoints) either by introducing new or malicious features (new datapoints) or disrupting the magnitude and positions of features in the input data. These adversarial perturbations exploit limitations in the manner in which features and patterns are learned by the ML model during training, thereby causing mispredictions with the ML model [129].

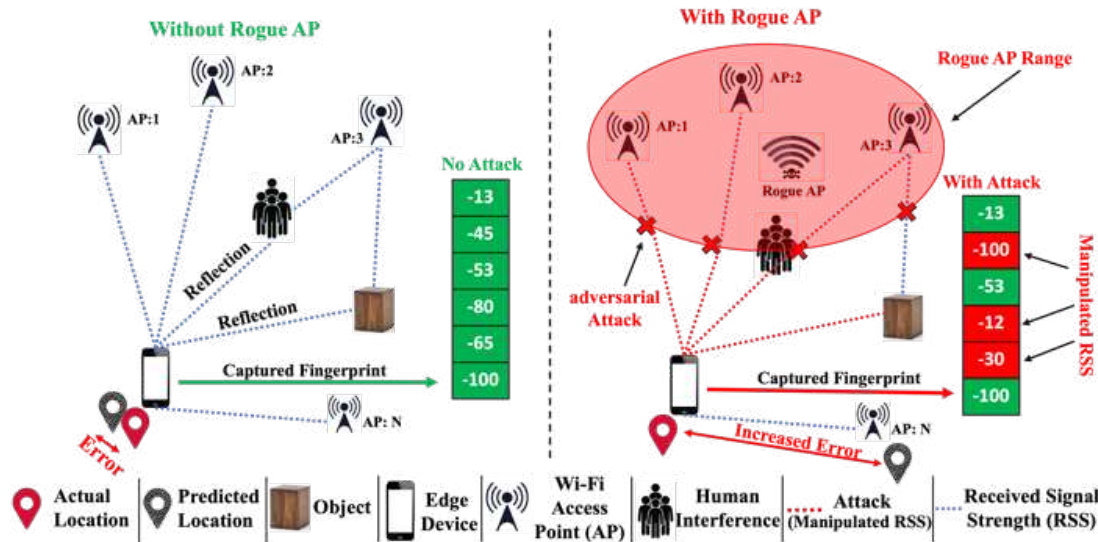


Figure 69: RSS fluctuations in indoor environment depicting real-world scenarios with and without the presence of rogue APs.

In the context of indoor localization, Wi-Fi RSS fingerprints are measured in decibels referenced to one milliwatt (dBm) and typically range from -100 dBm (weak signal) to 0 dBm

(strong signal). These fingerprints are very susceptible to fluctuations due to dynamic indoor environments and edge device heterogeneity, and perturbations due to adversarial attacks especially in the presence of rogue APs, as shown in Figure 69. Rogue APs can perturbate specific or all datapoints within an RSS fingerprint. This perturbed data may exhibit features characteristic of a different RP location, leading to increased prediction errors, as shown in Figure 69.

Rogue APs pose a threat to indoor localization systems by introducing deliberate perturbations through two distinct pathways: the transmitter side (involving APs) and the channel side (within the space between the AP and edge device).

- **Transmitter side:** This attack is executed from the transmitter side, specifically on the APs deployed in the indoor environment. The attack targets a legitimate AP in the environment, attempting to infect it with malicious data (malware). Once successful, the resulting rogue AP gains complete control over the legitimate AP, compromising the security of any operations performed by the legitimate AP. This poses a significant security risk, as the rogue AP can now manipulate RSS, leading to an increase in localization errors. This attack can compromise the robustness of the indoor localization solution in that environment.
- **Channel side:** This attack is executed from the channel side, specifically within the spatial domain between a legitimate AP and the edge device. The rogue AP monitors communication between the legitimate AP and edge devices and introduces carefully calibrated interference with the signals traveling through this space. Once successful, the rogue AP can manipulate the RSS captured by the edge device, that may mimic the characteristics of a different RP location. This manipulation

compromises the robustness of the indoor localization solution, as the altered RSS can lead to increase in localization errors.

8.4.1. ROGUE AP ATTACK IMPLEMENTATION

Rogue APs possess the capability to execute a variety of attacks. In the context of indoor localization, we delve into the practical implementation of these attacks, focusing on the transmitter side (evil twin attacks) and the channel side (man-in-the-middle attacks) within the indoor localization system. Notably, these attacks can be launched with minimal information about the target system, rendering them as grey box attacks. The nature of grey box attacks makes rogue APs an attractive choice for adversaries, as they do not require comprehensive knowledge of the indoor localization system. Unlike traditional white box attacks requiring complete access to privileged information, such as ML model architecture, localization framework policies, building layouts, and access to edge devices, grey box attacks operate with partial knowledge. This characteristic transforms rogue AP implementation into a more plug-and-play system for executing adversarial attacks. We next describe the two types of rogue AP attacks, illustrating their underlying methods and potential consequences.

- **Evil twin attacks:** This transmitter side rogue AP attack involves the creation of a malicious wireless network that mimics a legitimate one. The rogue AP utilizes malware to infect a legitimate AP, allowing it to gather critical information such as the SSID (service set identifier), MACID (media access control identifier), and other network parameters [131]. By replicating these parameters, the rogue AP tricks edge devices into connecting to it, masquerading as an authentic AP. Figure 70 demonstrates the implementation of the evil twin attack, which is

explored for the first time in the context of indoor localization, as part of this work. The rogue AP initiates the attack by targeting a legitimate Wi-Fi AP, mimicking its network parameters, and simultaneously blocking all communications from the legitimate Wi-Fi AP. Subsequently, the rogue AP broadcasts its own malicious Wi-Fi network (masquerading the authentic AP), that can inject malicious features into the RSS fingerprint collected by the edge device. These malicious features have the potential to falsify the edge device's perceived location, making it appear in a different location. This compromise in location information poses a severe threat to the entire indoor localization system.

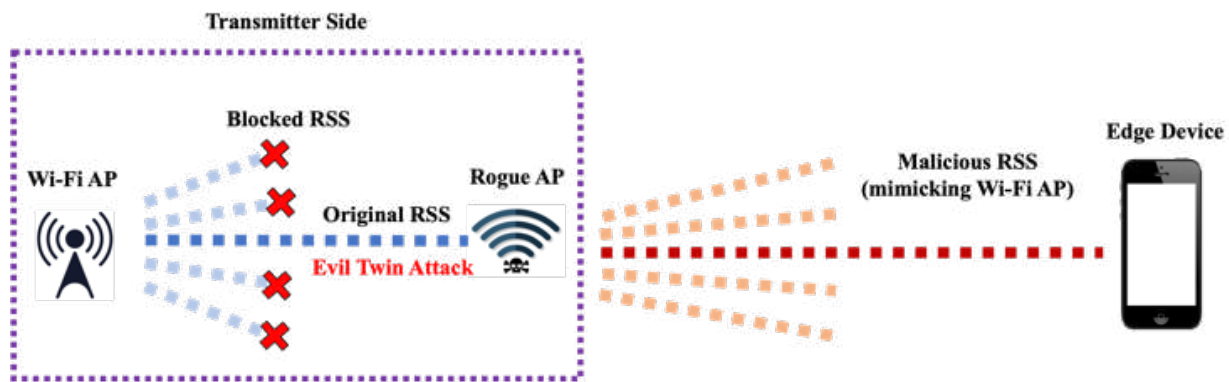


Figure 70: Evil twin attack during indoor localization.

- **Man-in-the-middle attacks:** This channel side rogue AP attack employs ARP (address resolution protocol) spoofing techniques to intercept communication between the legitimate Wi-Fi AP and the edge devices [132]. Operating within the spatial domain between the AP and the edge device, the rogue AP positions itself as an intermediary, intercepting signals transmitted between the legitimate AP and the edge device. Unlike direct communication, the man-in-the-middle attack allows the rogue AP to inspect, modify, or block the signals before relaying them to their intended destination. This interception provides the adversary with the

capability to alter RSS values in real-time, introducing discrepancies in the RSS features captured by the edge device. Figure 71 demonstrates the implementation of the man-in-the-middle attack for indoor localization.

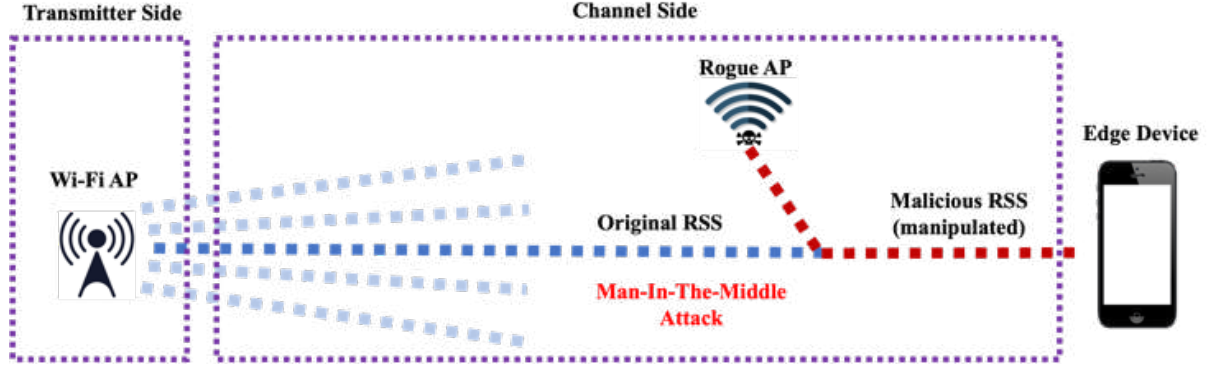


Figure 71: Man-in-the-middle attack during indoor localization.

8.4.2. ADVERSARIAL ATTACK METHODS

Adversarial perturbations, introduced by malicious entities, pose a threat to ML models, particularly in privacy-sensitive domains like indoor localization. The covert nature of these attacks necessitates robust mitigation methods. We identify and focus on three popular adversarial methods in this work: fast gradient sign method (FGSM) [129], projected gradient descent (PGD) [108], and momentum iterative method (MIM) [109]. Given the grey box nature of adversarial attacks (evil twin and man-in-the-middle attacks, discussed above), adversaries exploit minimal information about the localization framework. These methods introduce carefully calibrated perturbations into the RSS fingerprints using the ML model's loss function, making them a practical choice for studying the nuanced effects of adversarial attacks in indoor localization.

- **Fast gradient sign method (FGSM):** FGSM leverages the gradient information of the ML model's loss function with respect to the input data. This method perturbs the original input data by adding a small, controlled perturbations in the direction of the gradient sign. This intentional perturbation systematically alters both the magnitude and positions of features within the input data. In the context of indoor localization, perturbations induce changes in the values within the RSS fingerprint, effectively manipulating the magnitude and positions of features inherent in fingerprint data. Consequently, this perturbation can mislead the ML model by indicating features at a different RP location, thereby increasing errors in location predictions.

$$\eta = \epsilon * \text{sign}(\nabla J(\theta, X, Y)) \quad (32)$$

$$X_{Adv} = X + \eta \quad (33)$$

In the equations above, η represents the perturbations, θ represents the parameters of the ML model, and X and Y denote the RSS fingerprint and RP class, respectively. The hyperparameter ϵ controls the magnitude of the perturbation and $(\nabla J(\theta, X, Y))$ denotes the loss function of the ML model. X_{Adv} is the perturbed RSS data.

- **Projected gradient descent method (PGD):** PGD extends the concepts of FGSM by offering a more sophisticated approach in generating adversarial examples. PGD modifies FGSM by eliminating the sign function in Equation 32 and clipping the perturbations

between X and ϵ . While FGSM introduces perturbations in a single step, PGD refines the perturbation over multiple iterations $\{X_{Adv(0)}, X_{Adv(1)}, \dots, X_{Adv(N)}, X_{Adv(N+1)}\}$.

$$X_{Adv(0)} = X \quad (34)$$

$$\eta = Clip_{X,\epsilon} \left\{ \epsilon * \frac{\nabla J(\theta, X, Y)}{L|\nabla J(\theta, X, Y)|_2} \right\} \quad (35)$$

$$X_{Adv(N+1)} = X_{Adv(N)} + \eta \quad (36)$$

In Equation 34, X denotes the original input data and $X_{Adv(0)}$ denotes the perturbed adversarial sample at the initial iteration (0). Equation 35 computes perturbations η using a clipped function applied to the gradients of the loss function $\nabla J(\theta, X, Y)$ and $L|\nabla J(\theta, X, Y)|_2$ represents the squared L2 norm (ridge regularization) of the gradients of the loss function. This normalization step ensures that the perturbation is scaled appropriately, maintaining stability in generating the adversarial sample, while being clipped between X and ϵ (magnitude of the perturbation). These perturbations are added to $X_{Adv(N)}$ iteratively, as shown in Equation 36. This iterative refinement process enhances the potency of adversarial samples by introducing a more calibrated manipulation in feature magnitude and positions within the RSS fingerprint data, leading to more potent adversarial samples compared to FGSM.

- **Momentum iterative method (MIM):** MIM further refines the adversarial samples from PGD, by incorporating momentum into the perturbation generation process to enhance the efficiency of the perturbation search.

$$X_{Adv(N+1)} = Clip_{X,\epsilon}\{\alpha * X_{Adv(N)} + \eta\} \quad (37)$$

The perturbation η is calculated using Equation 35, similar to the PGD approach. In Equation 37, α is applied as momentum to the $X_{Adv(N)}$ of the previous iteration, while being clipped between X and ϵ (magnitude of the perturbation). By incorporating momentum into the perturbation generation process, MIM effectively manipulates RSS features and positions, leading to adversarial samples that induce more significant errors in the localization process, compared to FGSM and PGD. This enhanced perturbation poses substantial challenges to the robustness of indoor localization solutions.

8.4.3. ADVERSARIAL ATTACK FORMULATION FOR ML INDOOR LOCALIZATION

In formulating adversarial attacks for indoor localization systems, we employ the three distinctive methods discussed above: FGSM, PGD, and MIM. Our objective is to generate adversarial data by introducing perturbations that modify the features embedded within an RSS fingerprint. To generate potential real-world adversarial data effectively, we leverage two key parameters:

- **Perturbation strength (ϵ):** This crucial hyperparameter is used in FGSM, PGD, and MIM methods to introduce perturbations to the RSS fingerprints. In generating adversarial samples for indoor localization, we systematically adjust the ϵ value to encompass various perturbation strengths applicable in real-world scenarios. We vary ϵ from 0.1 to 0.5 to reflect a practical

perturbation scenario tailored for indoor localization [104]. This range is considered acceptable because it strikes a balance between being subtle enough to evade detection and significant enough to effectively test the system's robustness. Smaller values of ϵ (closer to 0.1) represent minor perturbations that are less likely to be noticed but might not challenge the system's defenses effectively, while larger values (up to 0.5) represent more noticeable perturbations that can more rigorously test the model's resilience. To emulate potential threats, rogue APs strategically deployed in indoor spaces can introduce these perturbations to mislead localization systems

- **Compromised APs (ϕ):** This parameter represents the quantity of legitimate APs that are subject to compromise by the rogue AP within the indoor system. In a typical scenario, rogue APs selectively attack a subset of legitimate APs. We utilize ϕ as a parameter to investigate the impact of the quantity of compromised APs on indoor localization performance. ϕ is set to range from 0 to 100, indicating the percentage of attacked APs, thus covering the spectrum from 0% to 100% of compromised APs. These attacked APs then introduce perturbations defined by the parameter ϵ .

8.5. SENTINEL FRAMEWORK: OVERVIEW

The SENTINEL framework consists of three key components: adversarial training, fingerprint image generation and the capsule neural network, as shown in Figure 72. The framework initiates in an offline phase, where RSS fingerprints are captured across different RPs within the building floorplan. Multiple fingerprints are collected per RP to effectively capture data variability. These fingerprints are labeled and stored in an RSS fingerprint database, forming the offline training data

for the SENTINEL framework. To fortify the framework against adversarial attacks, we employ an adversarial training mechanism (discussed in Section 8.5.1), which introduces adversarial samples derived from the RSS fingerprint database. This approach not only enhances the framework's resilience towards potential adversarial challenges but also introduces additional variability to the training data, thereby improving the framework's ability to enhance its resilience towards dynamic indoor environments and edge device heterogeneity in the online phase. Post-adversarial training, we transform both original (from RSS fingerprint database) and adversarial fingerprints into fingerprint images using the fingerprint image generation mechanism (discussed in Section 8.5.2), resulting in grayscale images. These grayscale images encapsulate crucial information about the indoor floorplan. The grayscale images then serve as input to the capsule neural network modified for the task at hand and carefully designed to address the spatial invariance problem in DL. The capsule neural network comprises of five sub-components: convolutional layer (CONV), primary capsule layer (PC), outer capsule layer (OC), an agreement-based routing algorithm, and the majority voting layer (all discussed in Section 8.5.3).

The domain-specific capsule neural network, once trained, is deployed on edge devices for predictions during the online phase. In the online phase, the edge devices (with the pre-trained ML model), scan for available RSS fingerprints at an unknown RP location. These received fingerprints are inherently susceptible to RSS fluctuations (introduced by dynamic environments and edge device heterogeneity) and potential adversarial attacks (introduced by rogue APs). Nevertheless, the SENTINEL framework is able to achieve resilience against these challenges, as discussed in Section 8.6 via sensitivity analysis experiments across devices and building, and comparison with state-of-the-art frameworks.

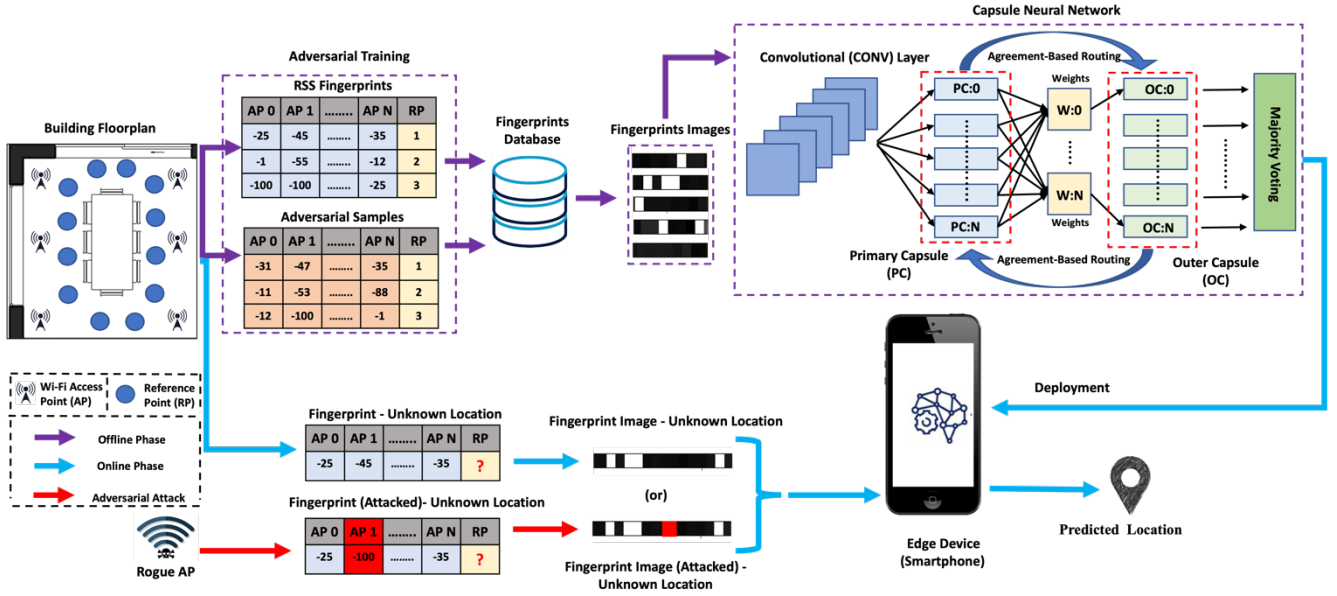


Figure 72: Overview of the SENTINEL framework, including the offline (training) phase and online (inference) phase.

8.5.1. ADVERSARIAL TRAINING MECHANISM

The SENTINEL framework enhances its resilience against adversarial attacks by implementing an adversarial training mechanism. This approach fortifies our capsule neural network by exposing it to a diverse mixture of adversarial and clean RSS examples during the training process. The fundamental concept behind adversarial training is to modify the loss function by incorporating adversarial examples, thereby rendering the capsule neural network resistant to adversarial attacks.

$$\nabla J(\theta, X, Y) = \nabla J(\theta, X, Y) + \nabla J(\theta, X + \eta, Y) \quad (38)$$

In Equation 38, η represents the perturbation introduced into the input data using different adversarial methods such as FGSM, PGD, and MIM, calculated using the gradients of the loss function (Equations 32, 34, and 36) with respect to the input data. In Section 8.6, we evaluate the performance of various adversarial training methods to assess SENTINEL’s efficacy in defending against adversarial attacks in the online phase.

8.5.2. FINGERPRINT IMAGE GENERATION

Post creation of the RSS fingerprint database (with clean + adversarial samples), the fingerprints are transformed into grayscale images to encapsulate crucial information about the indoor floorplan. Initially, the RSS fingerprints are arranged into matrices or tensors, with shape of (H, W) , where H represents the height (typically 1), and W signifies the width, representing the number of visible APs within the indoor environment. Each element in this tensor corresponds to the RSS measured by a specific AP at a particular RP. To convert these RSS fingerprint tensors into grayscale images, a mapping process is applied. This mapping function translates the RSS values into pixel intensities, ensuring that higher RSS values are represented with brighter pixels and lower RSS values with darker pixels. The resulting grayscale images have a shape of (N, H, W, C) , where N denotes the RPs, H represents the height (usually 1), W signifies the width (number of visible APs), and C represents the number of channels (typically 1 for grayscale). This conversion preserves the spatial information of RSS across the indoor space, facilitating effective localization.

8.5.3. CAPSULE NEURAL NETWORK ARCHITECTURE

The capsule neural network is a pivotal component of the SENTINEL framework, comprising of five sub-components: the convolutional (CONV) layer, primary capsule (PC) layer, outer capsule (OC) layer, an agreement-based routing algorithm, and a majority voting layer. The enhanced capsule neural network in SENTINEL possesses several key differences from EDGELOC [127] which uses a simple capsule neural network: 1) Unlike [127], SENTINEL integrates a majority voting layer to enhance prediction output, 2) Unlike [127], SENTINEL is tailored specifically for processing grayscale fingerprint images, 3) [127] targets device heterogeneity only, whereas SENTINEL optimizes hyperparameters differently to simultaneously target mitigation of dynamic environment induced RSS fluctuations, device heterogeneity, and adversarial attacks, and 4) SENTINEL is pruned in the number of capsules (both PC and OC layers) and neurons within each capsule, resulting in a more lightweight deployment on resource-constrained edge devices than [127] while maintaining accuracy. We compare SENTINEL against EDGELOC [127] in Section 8.6. In the rest of this section, we describe the various components of our SENTINEL capsule neural network.

- **Convolutional (CONV) layer:** The CONV layer captures spatial features within the grayscale fingerprint images. This layer employs convolutional filter kernels to extract distinctive patterns and features from the input images. Let us denote the grayscale RSS fingerprint image as IM , which has dimensions (N, H, W, C) . The convolutional layer consists of multiple filters kernels, denoted as F , which are applied to IM . The F slide across the entire IM , performing element-wise multiplications and summations, generating feature maps that highlight spatial features within the IM .

$$CON(p, q) = \sum_{i=0}^H \sum_{j=0}^W IM(p-i, q-j) * F(i, j) \quad (39)$$

In the equation above, $CON(p, q)$ denotes the feature at position (p, q) in the CONV feature map and $F(i, j)$ represents the corresponding element of the filter kernel. $IM(p-i, q-j)$ represents the pixel value of IM at position $(p-i, q-j)$. The summation is performed over the height (H) and width (W) of F . During training, the network learns the optimal values of F through backpropagation. This process enables the CONV layer to automatically detect and extract relevant spatial features from the input RSS fingerprint images, providing meaningful representations that contribute to the overall accuracy of the localization process.

- **Primary capsule (PC) layer:** The PC layer receives the spatial features extracted by the CONV layer and serves as the next processing stage in the capsule neural network. A capsule is defined as a group of neurons, where each capsule within the PC layer generates a vector, referred to as the “activity vector”. This vector captures both the magnitude (presence) and position of each feature in the RSS fingerprint. Unlike traditional neural networks (such as MLPs and CNNs) where neurons in subsequent layers are densely connected to all neurons in the preceding layer, the PC layer comprises of capsules, where each capsule corresponds to a specific spatial feature detected by the CONV layer. The activity vector $((u_{ij}))$ for capsule i is obtained through a series of computations:

$$S_i = \sum_j V_{ij} * CON_j \quad (40)$$

$$u_{ij} = \text{Squash}(S_i) = \frac{\|S_i\|^2}{1 + \|S_i\|^2} * \frac{S_i}{\|S_i\|} \quad (41)$$

In Equation 40, S_i represents the input for each capsule i , which is calculated as the weighted sum of outputs from the CONV layer using weight tensors (V_{ij}). These weight tensors determine the contribution of each feature from the CONV layer, enabling the PC layer to selectively focus on relevant spatial features. Subsequently, S_i is squashed using a non-linear activation function known as the squash function. The squash function transforms S_i into activity vectors u_{ij} , which represent the magnitude and position of the detected spatial features within the RSS fingerprint. This enables the PC layer to encode spatial relationships between features, enhancing the network's ability to capture meaningful representations of the indoor environment.

- **Outer capsule (OC) layer:** The OC layer performs classifications based on the activity vectors (u_{ij}), received from the PC layer. Each capsule in the OC layer corresponds to an RP class which determines the probability of the input fingerprint image belonging to that class. The classification process in the OC layer involves computing the agreement score between the u_{ij} and the weights tensors (W_{ij}) associated with each capsule in the OC layer.

$$a_i = u_{ij} * W_{ij} \quad (42)$$

$$P_i = \text{Softmax}(a_i) \quad (43)$$

In Equation 42, a_i represents the agreement score for capsule i . The W_{ij} contains the weight tensors associated with the connections between the PC and OC layers, determining the importance

of each spatial feature for the classification of the corresponding RP class. In Equation 43, P_i denotes the predicted RP of capsule i after applying the *Softmax* function to a_i from Equation 42. This function assigns probabilities to each RP class based on a_i , facilitating the classification process.

- Agreement-based routing algorithm:** The agreement-based routing algorithm plays a crucial role in refining the weight tensors (W_{ij}) between the PC and OC layers. After the OC layer receives activity vectors (u_{ij}) from the PC layer, the agreement scores (a_i) are computed using Equation 42, representing the agreement between the u_{ij} and W_{ij} associated with each capsule in the OC layer. The goal of the routing algorithm is to iteratively adjust these weights tensors based on the a_i achieved. The routing process involves several iterative steps, where a_i are used to update the W_{ij} in a way that maximizes agreement between the a_i and the predicted RP classes. This iterative refinement enhances the network's ability to accurately classify input fingerprint images.
- Majority voting layer:** The majority voting layer is the final component of the proposed capsule neural network. This layer aggregates the predictions (P_i) generated by the OC layer for each capsule. The majority voting mechanism aims to determine the final prediction by selecting the RP class with the highest number of aligned predictions from the capsules in the OC layer.

$$Prediction = Argmax (P_0, P_1, \dots, P_n) \quad (44)$$

In Equation 44, n represents the total number of RP classes. The *Argmax* function selects the RP class with the highest probability as the final prediction. By ensuring that a majority of capsules agree on the final class, the majority voting layer reduces the impact of erroneous predictions from individual capsules.

8.6. EXPERIMENTAL SETUP

In this section, we describe our experimental setup, designed to evaluate the performance of our proposed SENTINEL framework in real-world scenarios. Our objective is to conduct comprehensive comparisons with state-of-the-art indoor localization frameworks, including CNNLOC [28], VITAL [80], EDGELOC [127], ADVLOC [130], and CALLOC [87], using simulated (FGSM, PGD, and MIM) and real-world RSSRogueLoc [59] data. To ensure the robustness of our evaluation, we embarked on an extensive data collection process. This involved gathering RSS fingerprints from diverse devices commonly available to the public, to capture performance across real-world scenarios. Data was collected during regular working hours, incorporating both dynamic and static occupants to reflect realistic conditions. Table 14 shows an overview of the real devices utilized in our experiments.

Table 14: Devices used to collect RSS fingerprints

Device Name	Wi-Fi Chipset	Acronym	Year
BLU Vivo 8	MediaTek Helio P10	BLU	2017
Google Pixel 6a	Google Tensor G1	GOOGLE	2022
HTC U11	Qualcomm Snapdragon 835	HTC	2017
Motorola Z2	Qualcomm Snapdragon 835	MOTO	2017
Nokia 7.1	Qualcomm Snapdragon 636	NOKIA	2018
OnePlus Nord 200	Qualcomm Snapdragon 480	ONEPLUS	2021
Xiaomi Redmi 10A	MediaTek Helio G88	REDMI	2022
Samsung A14	Samsung Exynos 850	SAMSUNG	2023

To ensure a comprehensive evaluation across diverse environmental conditions, we select building floorplans with varying factors such as path length, the number of visible APs, and environmental noise characteristics, as shown in Figure 73. Our data collection strategy is designed to facilitate thorough training and testing of the SENTINEL framework. For each building floorplan, we allocate five fingerprints per RP for training and one fingerprint per RP, per device, and per building, for testing. Acknowledging the substantial effort required to gather a large volume of offline training data, we restrict the collection of offline data to a single device. To facilitate this, we designate the MOTO device as the primary training device. This approach aims to balance the need for sufficient training data with practical considerations regarding data collection efforts. All devices in Table 14 are used in the online phase during testing.

The SENTINEL framework is configured with specific architectural hyperparameters. The CONV layer is equipped with 32 filters and the PC layer comprises of 8 capsules with each capsule containing a dimension of 32 neurons. Furthermore, the OC layer contains capsules equal to the number of RP classes with a dimension of 32 neurons each, trained over 300 epochs using the Adam optimizer (learning rate = 0.001) and the sparse categorical cross-entropy loss function. The capsule neural network architecture results in a total of 2,117,687 trainable parameters, with a compact model size of 8.07 MB, facilitating low overhead deployment on most resource-constrained edge devices. Additionally, the SENTINEL framework incorporates an adversarial training mechanism aimed at enhancing its resilience against potential adversarial attacks. Adversarial samples are generated using the FGSM, PGD, and MIM approaches with ϵ set to 0.1 and ϕ set to 100% (for training only). Each variant of our trained capsule neural network, augmented with adversarial samples, is denoted with a suffix. For instance, the model trained without adversarial samples is referred to as SENTINEL-NONE, while models trained with

FGSM, PGD and MIM samples are labeled SENTINEL-FGSM, SENTINEL-PGD and SENTINEL-MIM, respectively.

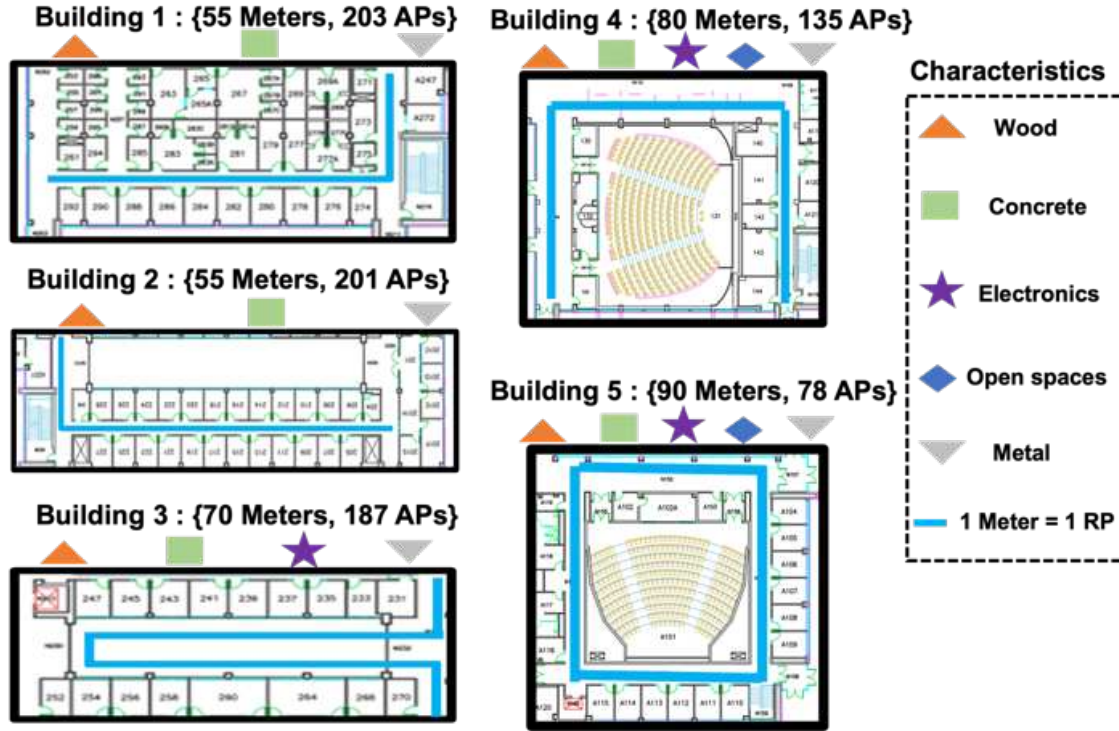


Figure 73: Building floorplan layouts with varying path length, visible APs, and characteristics.

8.6.1. EFFECTS OF ADVERSARIAL TRAINING ON HETEROGENEITY

In this section, we evaluate the performance of the SENTINEL framework under various adversarial training scenarios (FGSM, PGD, and MIM), separately. This experiment focuses on understanding the framework's efficacy across diverse building floorplans and its response to device heterogeneity. In Figure 74, we present heatmaps depicting the performance of the four SENTINEL variants: SENTINEL-NONE, SENTINEL-FGSM, SENTINEL-PGD, and SENTINEL-MIM. These models are individually trained on data collected exclusively from a single device (MOTO) and incorporate their respective adversarial training techniques.

SENTINEL-NONE is trained without including any adversarial samples, providing a comparison of the effects of including adversarial training to the SENTINEL framework. Evaluation of these model variants are conducted using data acquired from all eight available devices across the five building floorplans, without any adversarial interference.

In Figure 74, the X-axis of each heatmap represents the testing devices, while the Y-axis corresponds to the different buildings used for evaluation. Each cell within the heatmap indicates the average prediction error (in meters) across all RPs for a specific combination of test device and building floorplan. We observe differences in prediction errors across all the SENTINEL variants, due to the differences in adversarial training methods used. We note an increase in prediction errors when going from building 1 to 5, which can be attributed to increasing environmental dynamic causing higher variations in the selected building paths. For instance, building 1 exhibited low environmental noise, likely due to fewer people moving along the path during the testing. It also had relatively shorter path lengths, which overall resulted in lower prediction errors. In contrast, building 5 experienced higher environmental noise due to significantly more people moving along the path during the testing phase, and longer path lengths, leading to higher prediction errors.

SENTINEL-FGSM consistently exhibits the lowest prediction errors, followed by SENTINEL-PGD, SENTINEL-NONE and SENTINEL-MIM. This trend suggests that while more advanced adversarial training methods like PGD and MIM may offer refined perturbations, they also introduce complexity and potential instability during training, leading to overfitting. The overfitting occurs because the adversarial samples generated by PGD and MIM involve multiple iterations of perturbations, making them more complex and causing feature mismatches between RP classes. As a result, the model may become overly specialized to these adversarial examples, reducing its ability to generalize well to unseen, real-world data. SENTINEL-FGSM however,

stands out due to its balance between perturbation effectiveness and model stability. Its non-iterative nature allows for smaller, controlled perturbations, reducing the chances of a feature mismatch between legitimate and FGSM samples.

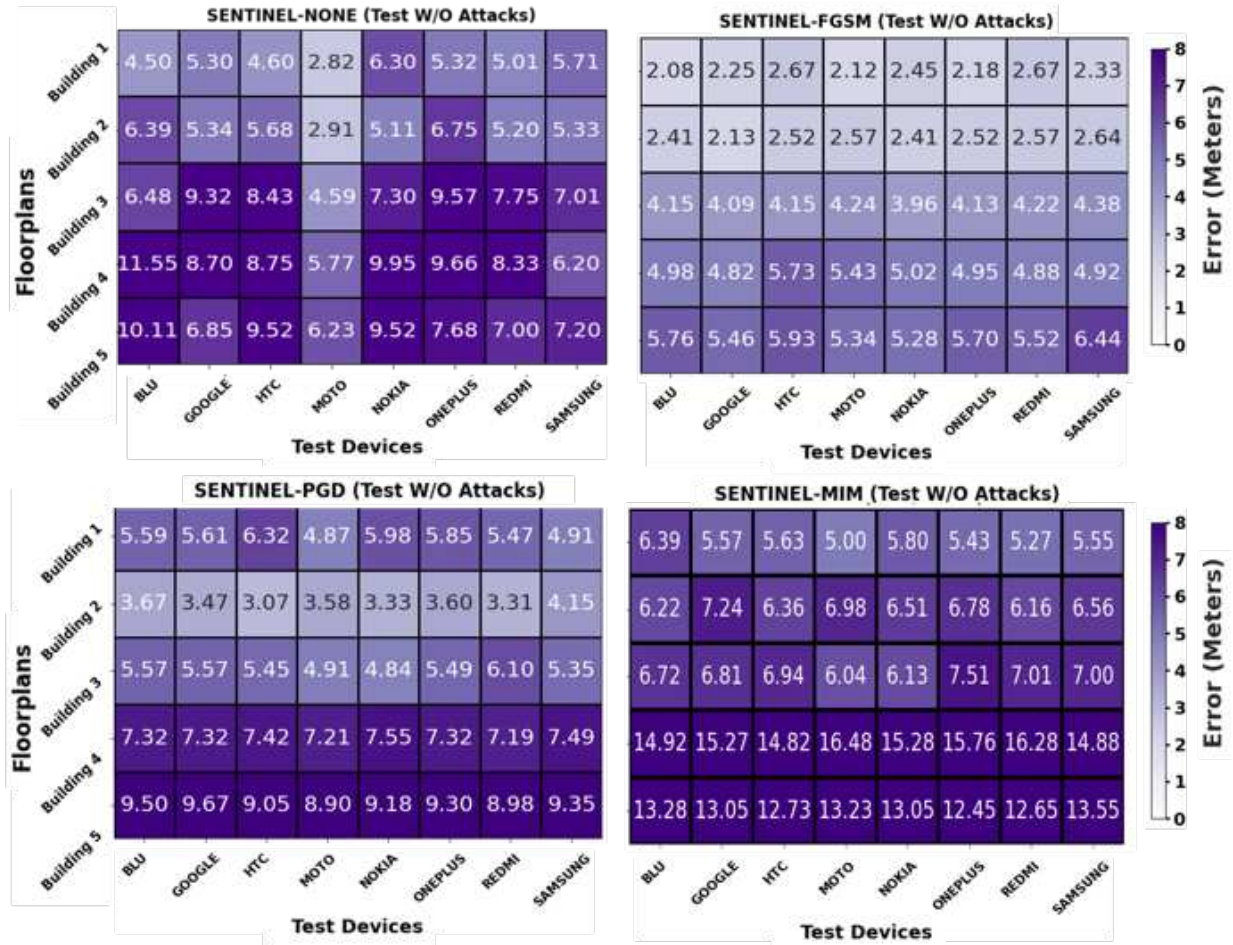


Figure 74: Performance of the SENTINEL variants across different devices and building floorplans.

To illustrate the impact of device heterogeneity and assess the performance of the SENTINEL variants, we present Figure 75 more clearly. Here, the X-axis represents the testing devices, and the Y-axis denotes the prediction error in meters. Each bar represents the average prediction error per device across all building floorplans, with error bars included to indicate the range of errors

observed per testing device, with the lower whisker representing the best case and the upper whisker representing the worst-case location error. In Figure 75, we observe that the average error per testing device remains consistent for each SENTINEL variant. However, the SENTINEL-NONE variant exhibits the least consistency in prediction errors across the testing devices, with some devices showing higher errors while others show lower errors. This suggests lower resilience to heterogeneity for the SENTINEL-NONE variant. Conversely, other SENTINEL variants show consistent prediction errors regardless of the training or testing devices used, indicating better heterogeneity resilience. Furthermore, incorporating adversarial training not only strengthens the robustness of the SENTINEL variants against adversarial attacks but also improves their resilience to heterogeneity. By subjecting the models to adversarial perturbations during training, the variants learn more generalized features, making them less sensitive to fluctuations from the testing devices. Particularly noteworthy is the performance of SENTINEL-FGSM, with up to $1.48\times$ to $2.43\times$ lower average and worst-case errors compared to the rest of the SENTINEL variants.

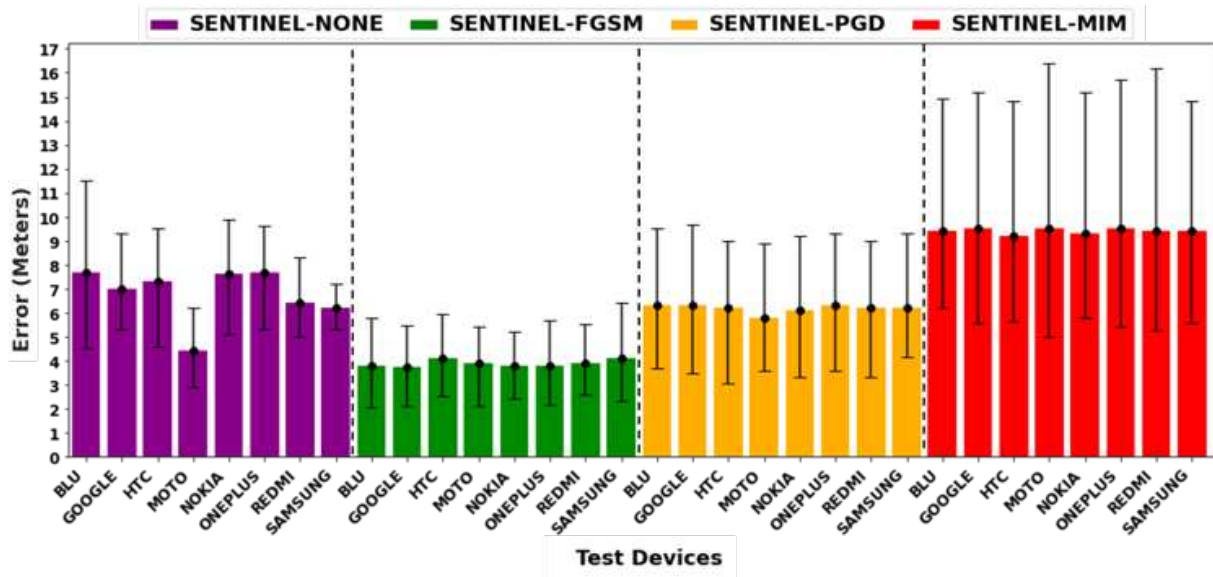


Figure 75: Performance summary for SENTINEL variants.

8.6.2. EVALUATING THE IMPACT OF VARYING COMPROMISED APS (Φ)

In this section, we investigate the impact of varying ϕ in the testing phase, using different adversarial attack methods (FGSM, PGD, MIM), on the performance of the SENTINEL variants. To maintain consistency, we set the attack ϵ to 0.1, indicating 10% added perturbations per ϕ . In Figure 76, the X-axis represents ϕ , ranging from 0 (no attacked APs) to 100 (all visible APs being attacked). The Y-axis denotes prediction errors measured in meters and the line plots illustrate the performance of each SENTINEL variant under the three adversarial attack methods. In Figure 76, each marker indicates the average prediction error across all testing devices and building floorplans.

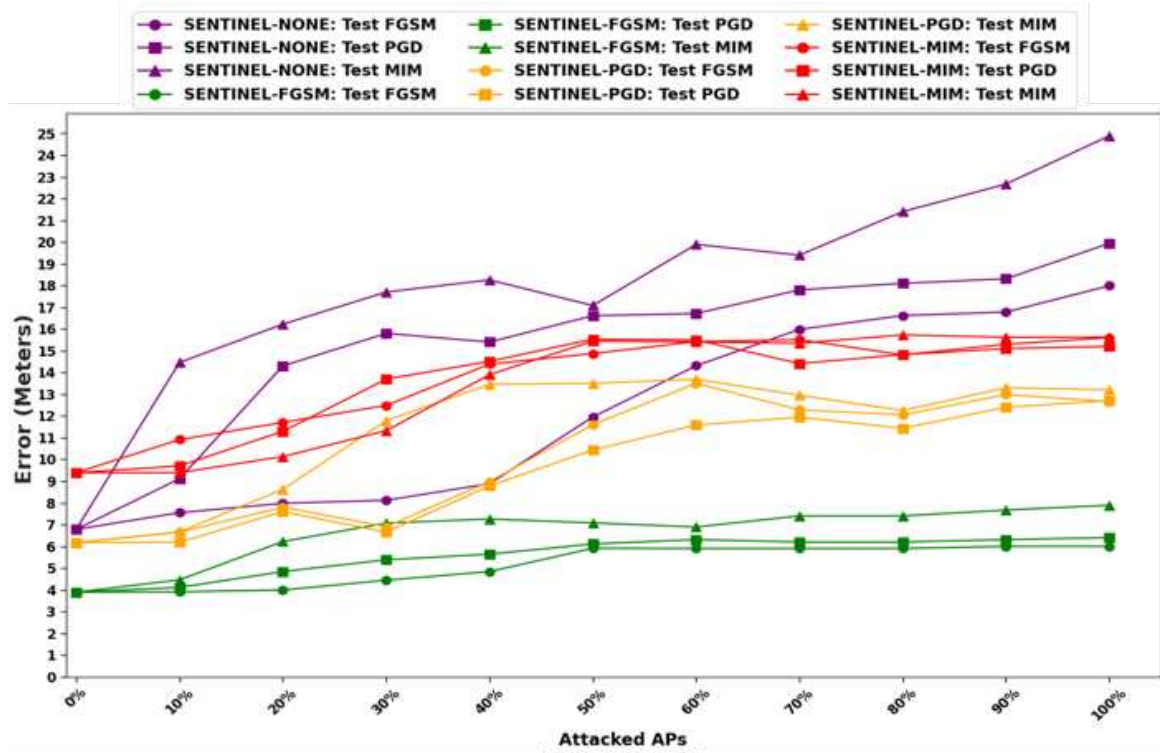


Figure 76: Performance of the four SENTINEL variants on simulated adversarial attacks through varying ϕ .

We observe that as ϕ increases, the prediction errors for all SENTINEL variants also increase. However, there is a stabilization point observed at approximately $\phi = 50\%$ for most variants methods (except SENTINEL-NONE, which lacks adversarial training), suggesting that the performance of the SENTINEL variants remains relatively unaffected when a significant portion of APs are compromised. This stabilization point indicates that the SENTINEL variants are resilient to attacks involving large numbers of compromised APs. Additionally, most variants demonstrate resilience against various adversarial attack methods (except SENTINEL-NONE), as evidenced by the almost flat line in prediction errors. Specifically, when subjected to the FGSM attack, the SENTINEL-FGSM model exhibits 1.90 $\times$, 2.35 $\times$, and 2.64 $\times$ lower average errors compared to the SENTINEL-PGD, SENTINEL-NONE and SENTINEL-MIM models, respectively. Similarly, under the PGD attack, the SENTINEL-FGSM model demonstrates 1.69 $\times$, 2.75 $\times$, and 2.40 $\times$ lower average errors compared to the SENTINEL-PGD, SENTINEL-NONE and SENTINEL-MIM models, respectively. Lastly, when influenced by the MIM attack, the SENTINEL-FGSM model shows 1.67 $\times$, 2.71 $\times$, and 2.15 $\times$ lower average errors compared to the SENTINEL-PGD, SENTINEL-NONE and SENTINEL-MIM models, respectively.

8.6.3. EVALUATING THE IMPACT OF VARYING PERTURBATIONS (ϵ)

In this section, we explore the impact of varying levels of perturbation strength (ϵ) in the testing phase on the performance of all SENTINEL variants. Our objective is to investigate how the prediction performance of each SENTINEL variant is affected by changes in ϵ , ranging from 0 (indicating no attack) to 0.5 (representing a 50% increase in added perturbations). In Figure 77, the X-axis represents the varying levels of ϵ , while the Y-axis denotes the prediction error in

meters. Each bar in the plot signifies the average prediction error across all testing devices, building floorplans, and ϕ values. Additionally, error bars are included to depict the range between the best (lower whisker) and worst-case (upper whisker) prediction errors. Our analysis reveals that as ϵ increases, there is a slight rise in prediction errors. However, we observe that all SENTINEL variants stabilize at approximately $\epsilon = 0.2$ (except SENTINEL-NONE, lacking adversarial training). This suggests that regardless of the increase in perturbation strength, all SENTINEL models demonstrate consistent performance. Furthermore, we observe that the SENTINEL-FGSM variant consistently outperforms all other SENTINEL variants. On average, SENTINEL-FGSM demonstrates $1.48\times$, $2.81\times$, and $1.90\times$ lower average prediction errors compared to SENTINEL-PGD, SENTINEL-NONE and SENTINEL-MIM, respectively. The superior performance of the SENTINEL-FGSM variant, even as ϵ increases during testing, can be attributed to the robustness gained through FGSM-based adversarial training. Although the model was trained with a fixed ϵ value, the adversarial training process encourages the model to capture underlying patterns in feature positions that are susceptible to adversarial attacks. This enables the model to generalize and adapt to perturbations even on varying ϵ . In contrast, other methods like PGD and MIM often induce significant perturbations in underlying features, leading to overfitting and reduced resilience during testing. The chosen epsilon range of 0 to 0.5 represents a practical attack range for indoor localization [104].

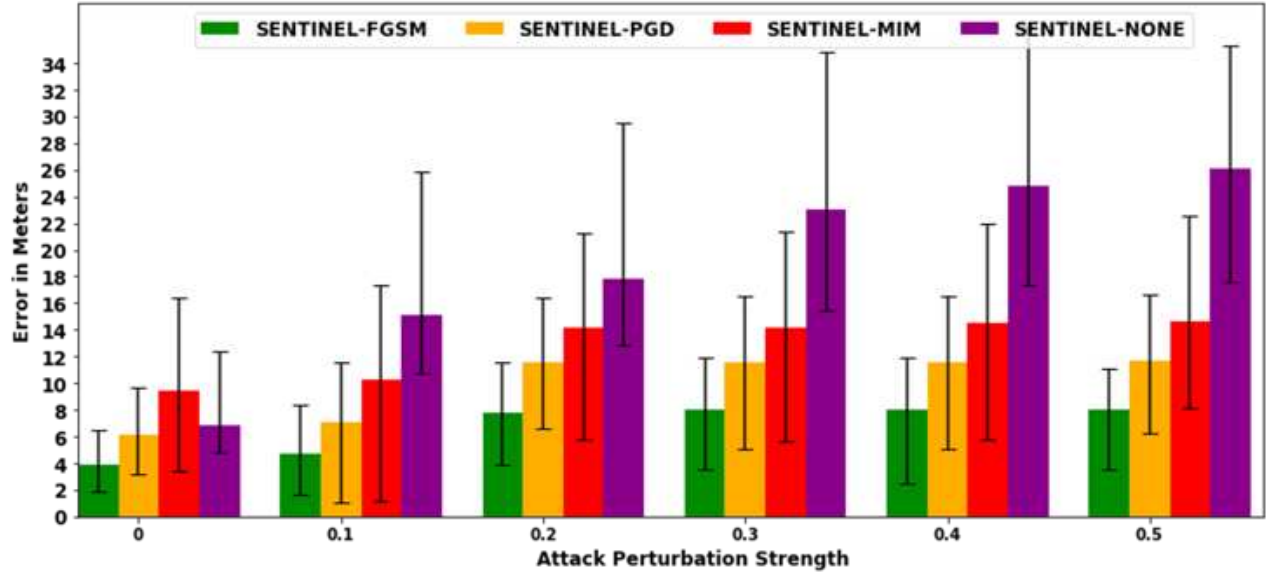


Figure 77: Performance of the three SENTINEL variants on simulated adversarial attacks through varying ϵ .

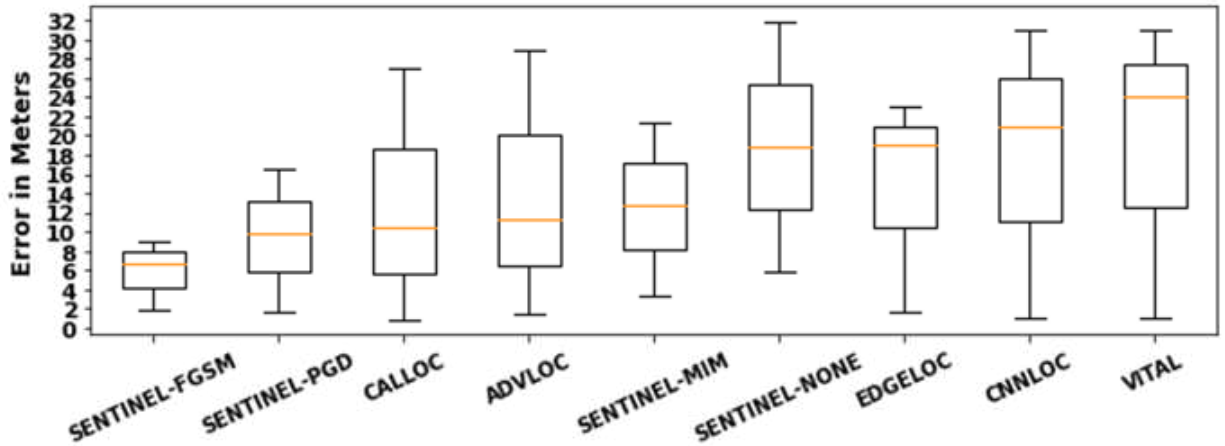


Figure 78: Performance comparisons of all SENTINEL variants against state-of-the-art indoor localization frameworks.

8.6.4. COMPARISON AGAINST STATE-OF-THE-ART FRAMEWORKS

In this section, we compare the performance of all SENTINEL variants against state-of-the-art indoor localization frameworks across various parameters including different devices, building

floorplans, ϵ (ranging from 0 to 0.5), and ϕ (ranging from 0 to 100). Figure 78 presents a box and whiskers plot, showcasing the comparison of the best case (lower whisker), worst case (upper whisker), and average (orange line) errors across all frameworks. This enhanced resilience can be attributed to the adversarial training and capsule neural network employed by the SENTINEL framework. The FGSM-based adversarial training introduces optimal adversarial features and feature dispositions (magnitude and positions), contrasting with other adversarial training methods that may lead to overfitting. The proposed capsule neural network treats each feature as a vector, effectively recognizing and capturing underlying patterns between the original (clean) and adversarial samples during training. This enables the SENTINEL-FGSM model to demonstrate lower prediction errors across various scenarios and metrics compared to the other frameworks. The SENTINEL-FGSM model demonstrates 1.47 $\times$, 1.55 $\times$, 1.68 $\times$, 1.91 $\times$, 2.82 $\times$, 2.83 $\times$, 3.13 $\times$, and 3.5 $\times$ lower average errors compared to SENTINEL-PGD, CALLOC, ADVLOC, SENTINEL-MIM, SENTINEL-NONE, EDGELOC, CNNLOC, and VITAL, respectively. Additionally, recognizing the need for lightweight frameworks adaptable for resource-constrained edge devices, we analyze the parameter count and memory footprint of the various frameworks as shown in Table 15. SENTINEL yields a compact model size of 8.07 MB.

Table 15: Model parameters, size of all frameworks

Framework	Total Parameters	Model Size
CALLOC	652,390	2.48 MB
CNNLOC	858,720	3.27 MB
ADVLOC	1,746,752	6.99 MB
SENTINEL	2,117,687	8.07 MB
EDGELOC	2,317,687	8.84 MB
VITAL	2,347,006	8.95 MB

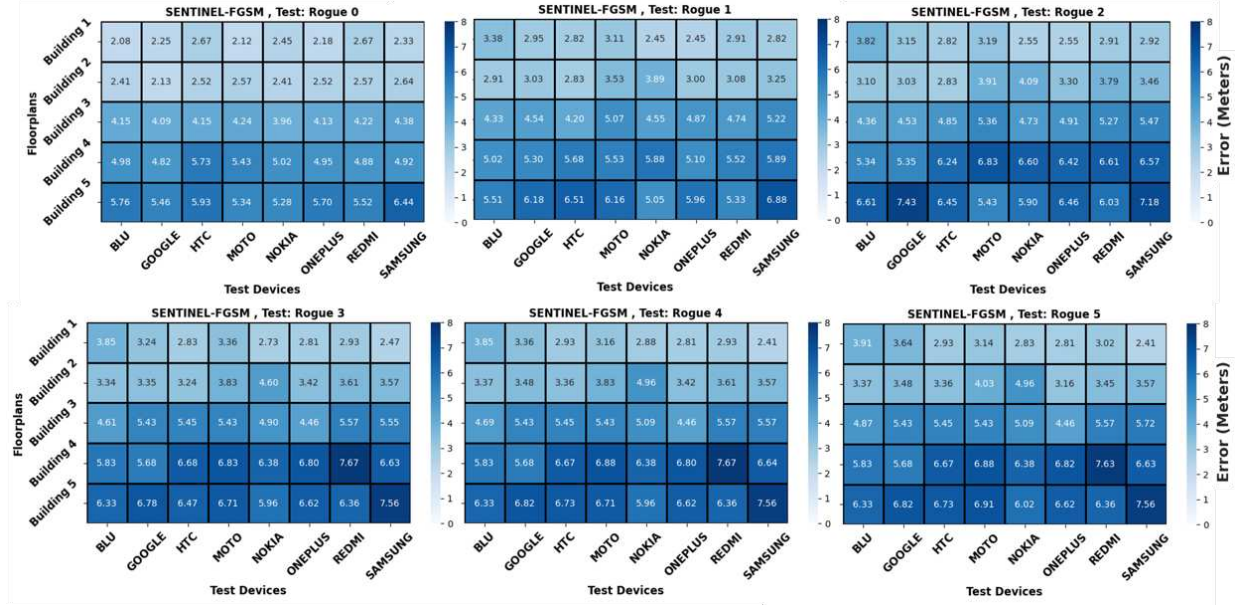


Figure 79: Device-Floorplan wise performance of SENTINEL-FGSM on RSSRogueLoc dataset.

8.6.5. EVALUATION ON THE NEW REAL-WORLD ROGUE AP ATTACK DATASET

In this section, we introduce a novel Wi-Fi RSS fingerprint dataset named RSSRogueLoc [59], designed to capture the detrimental effects of rogue APs for indoor localization systems. Unlike prior works which primarily rely on simulated adversarial attacks introduced by methods such as FGSM, PGD, and MIM, RSSRogueLoc delves into real-world adversarial scenarios, particularly those involving rogue APs. Building on the dataset outlined in Section 8.6.1, RSSRogueLoc introduces a secondary testing dataset comprising up to five new devices configured as rogue APs (devices detailed in Table 16), designed to execute evil twin attacks as discussed in Section 8.4.1, where each rogue is configured to impact one legitimate AP. The RSSRogueLoc fingerprints were collected by incrementally introducing rogue APs across all RPs within each building floorplan. This sequential escalation started from Rogue 0, signifying the absence of all rogues, followed by Rogue 1 with one rogue per RP per floorplan, Rogue 2 with two rogues per RP per floorplan, Rogue 3 with three rogues per RP per floorplan, Rogue 4 with four rogues per RP per floorplan,

and finally Rogue 5 with five rogues per RP per floorplan. The testing fingerprints were collected using all eight devices mentioned in Table 14. This process unfolded over several weeks, to thoroughly capture the complexities of rogue AP configurations across numerous RPs and building floorplans. To further facilitate research in this nascent field, we will open-source the RSSRogueLoc [59] dataset, to allow the indoor localization community to explore the impact of real-world rogue APs.

Table 16 : Rogue AP devices used in rssrogueloc

Device Name	Wi-Fi Chipset	Device Type
Samsung G991U	Samsung Exynos 2100	Smartphone
Apple A2789	Apple U2	Laptop
HP 840 G6	Intel Wi-Fi AX201	Laptop
Vivo V2025	Qualcomm Snapdragon 720G	Smartphone
HP 840 G10	Intel Wi-Fi AX211	Laptop

We evaluate the performance of the best-performing SENTINEL variant (SENTINEL-FGSM), on the newly introduced RSSRogueLoc dataset. In Figure 79, the X-axis of each heatmap represents the testing devices, while the Y-axis corresponds to the building floorplans. Each cell within the heatmap indicates the average prediction error (in meters) across all RPs. The heatmaps present comprehensive results for each of the Rogue configurations tested on the SENTINEL-FGSM model. Notably, we observe minimal changes in errors even with an increase in the number of rogues in the respective building floorplans, suggesting that the SENTINEL-FGSM model effectively addresses adversarial attacks posed by rogue APs. To provide additional insights into the performance of all SENTINEL variants and state-of-the-art baseline frameworks on the RSSRogueLoc dataset, we present Figure 80. The SENTINEL-FGSM model demonstrates $1.51\times$, $1.65\times$, $1.68\times$, $1.91\times$, $2.04\times$, $2.27\times$, $2.34\times$, and $2.80\times$ lower average error compared to SENTINEL-

PGD, CALLOC, ADVLOC, EDGELOC, SENTINEL-MIM, SENTINEL-NONE, CNNLOC, and VITAL, respectively.

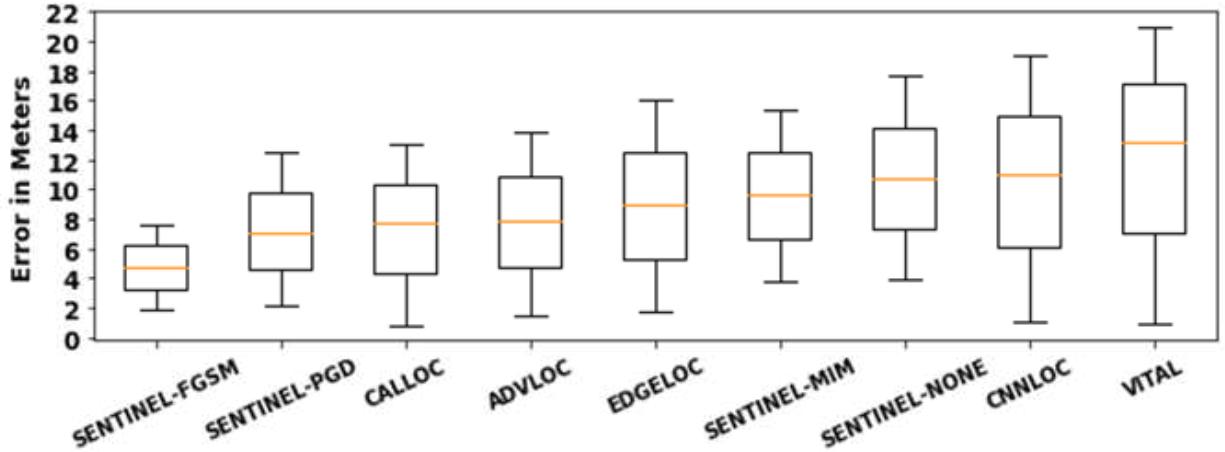


Figure 80: Performance comparisons of all SENTINEL models against state-of-the-art on the RSSRogueLoc dataset.

8.7. CONCLUSION

The SENTINEL framework proposed in this work exhibits resilience against RSS fluctuations arising from environmental noise, edge device heterogeneity, and challenging adversarial attacks, due to its novel combination of adversarial training and modified capsule neural networks, while being relatively lightweight for edge device deployment. Through rigorous evaluation, we found that the SENTINEL-FGSM variant consistently achieves the lowest indoor localization errors, outperforming all baseline frameworks by $1.47\times$ to $3.5\times$ in average errors and $1.83\times$ to $3.4\times$ in worst-case errors on simulated adversarial attacks. Moreover, our introduction of the RSSRogueLoc dataset, designed to capture real-world effects of rogue APs (performing evil twin attacks in real-time), further highlights the superiority of the SENTINEL-FGSM variant. With

1.51 $\times$ to 2.8 $\times$ lower average errors and 1.63 $\times$ to 2.74 $\times$ lower worst-case errors compared to other state-of-the-art frameworks.

CHAPTER 9

SAFELOC: OVERCOMING DATA POISONING ATTACKS IN HETEROGENEOUS FEDERATED MACHINE LEARNING FOR INDOOR LOCALIZATION

Machine learning (ML) based indoor localization solutions are critical for many emerging applications, yet their efficacy is often compromised by hardware/software variations across mobile devices (i.e., device heterogeneity) and the threat of ML data poisoning attacks. Conventional methods aimed at countering these challenges show limited resilience to the uncertainties created by these phenomena. In response, we introduce SAFELOC, a novel framework that not only minimizes localization errors under these challenging conditions but also ensures model compactness for efficient mobile device deployment. SAFELOC introduces a novel fused neural network architecture that performs data poisoning detection and localization, with a low model footprint using federated learning (FL). Additionally, a dynamic saliency map-based aggregation strategy is designed to adapt based on the severity of the detected data poisoning scenario. Experimental evaluations demonstrate that SAFELOC achieves improvements of up to $5.9\times$ in mean localization error, $7.8\times$ in worst-case localization error, and a $2.1\times$ reduction in model inference latency compared to state-of-the-art indoor localization frameworks across diverse indoor environments and data poisoning attack scenarios.

9.1. INTRODUCTION

Indoor localization systems are rapidly gaining traction across the domains of augmented and virtual reality (AR/VR), smart home automation, asset tracking, indoor navigation, and indoor drone technologies [170]. Many emerging advancements in these domains require accurate, lightweight, and real-time location tracking on resource-limited edge and mobile devices [64].

More broadly, the indoor positioning and navigation market that was valued at USD 10.9 billion in 2023 is expected to reach USD 29.8 billion by 2028, underscoring the critical role of these systems [171]. One of the most widely used methods for indoor localization is based on Wi-Fi received signal strength (RSS) fingerprinting, which utilizes the strength of signals from Wi-Fi access points (APs) to determine the position of mobile devices indoors [172]. This approach benefits from the ubiquity of Wi-Fi infrastructure and its compatibility with modern smart mobile devices, making it a favorable solution for real-world deployments.

Wi-Fi RSS fingerprinting operates by collecting RSS values from available APs at specific locations, forming a "fingerprint" unique to that position. As mobile devices move across different reference points (RPs) within a building, these RSS values fluctuate due to environmental factors and changes in position [172], [135]. Machine learning (ML) models are trained on these patterns, allowing them to map RSS fingerprints to precise locations. However, device heterogeneity—the differences in hardware, software, and firmware across mobile devices—introduces additional variability in the RSS measurements, complicating the localization process [135]. Furthermore, environmental factors such as multipath fading, shadowing, and human interference exacerbate this variability, making it increasingly challenging for ML models to maintain accurate predictions [134], [184]. A traditional solution to address these challenges involves extensive data collection across a wide variety of devices and environmental conditions to ensure that the ML model learns to handle this variability [64], [87], [172], [183]. However, such a comprehensive data collection process is resource-intensive and impractical in real-world scenarios. A more efficient approach is federated learning (FL), which allows user-carried mobile devices (referred to as clients) to train local models (LMs) using their collected RSS fingerprints and collaboratively update a global model (GM) at a central server [173]. This approach helps FL systems adapt to heterogeneous

environments and devices while reducing manual data collection efforts and maintaining user privacy by ensuring that sensitive data remains local to the client devices.

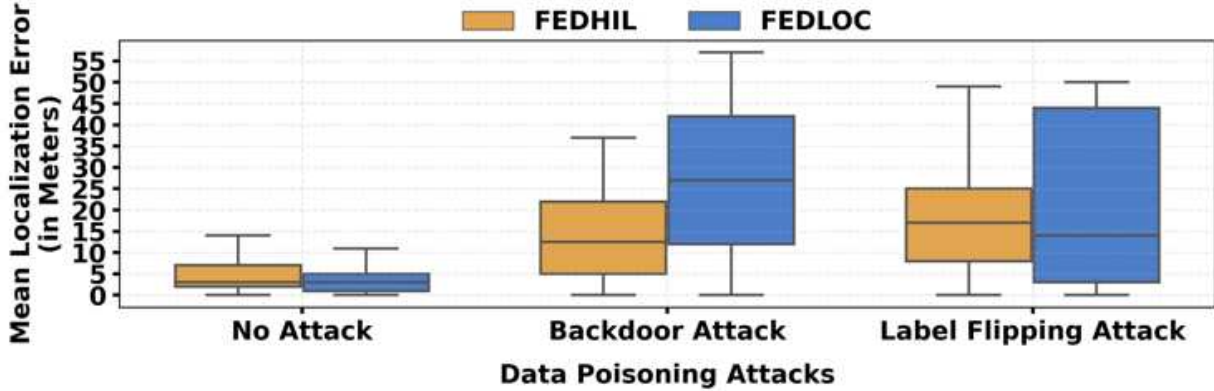


Figure 81: Indoor localization error comparison for FEDLOC [175] and FEDHIL [88] under label flipping and backdoor poisoning attacks.

Despite these advantages, FL-based indoor localization systems are vulnerable to data poisoning attacks, where malicious clients intentionally inject poisoned or manipulated data into the GM’s learning process [174]. Such attacks severely degrade the performance of the GM, as the erroneous LM updates from compromised devices are aggregated into the GM, leading to inaccurate localization predictions. To illustrate the impact of data poisoning attacks on FL-based indoor localization, we conducted experiments on two state-of-the-art FL-based indoor localization frameworks, FEDHIL [88] and FEDLOC [175], both of which use a three-layer deep neural network (DNN) as their GM. However, the aggregation strategies they employ differ: FEDLOC uses the traditional federated averaging (FEDAvg) method [176] to aggregate the LM updates, while FEDHIL utilizes a domain-specific selective weight aggregation technique that averages only specific weight tensors to mitigate bias from individual clients (discussed further in Section 9.3). We subjected both frameworks to two types of data poisoning attacks: label flipping,

where the labels are randomly altered, and backdoor attacks, implemented using the popular Fast Gradient Sign Method (FGSM) [107] (discussed further in Section 9.4).

In Figure 81, we present the performance degradation caused by these attacks, showing worst case errors (upper whisker), best case errors (lower whisker), and mean errors (center bar). Under label flipping, FEDLOC exhibits a $3.5\times$ increase in mean localization errors, while backdoor attacks cause a staggering $6.5\times$ rise in mean errors. FEDHIL, due to its selective aggregation strategy, shows relatively better resilience to backdoor attacks ($3.25\times$ increase in mean errors) but slightly worse performance for label flipping attacks ($3.9\times$ increase in mean errors). To some extent, the results highlight the role of aggregation strategies in defending against data poisoning attacks in FL systems, but in general both frameworks suffer from high localization errors. Moreover, many existing solutions for mitigating data poisoning rely on bulky, resource-intensive models that are unsuitable for real-time deployment on resource-limited edge and mobile devices (see Section 9.3), where lightweight models and low inference latency are critical requirements.

9.2. CONTRIBUTIONS

To address these multi-faceted challenges, we propose SAFELOC, a novel FL-based indoor localization framework that mitigates uncertainties due to client device heterogeneity and ML data poisoning attacks, while ensuring lightweight deployment on mobile devices. Our novel contributions are:

- We propose a new fused neural network architecture that performs poison detection, poison de-noising, and localization within the same model, and is optimized for efficient deployment on resource-limited devices.

- We propose a novel saliency map-based aggregation strategy that adjusts the influence of LMs based on the severity of detected poisoning, ensuring robust protection against ML data poisoning attacks.
- We perform extensive analysis on real-world data to evaluate SAFELOC under different data poisoning attacks and heterogeneous mobile devices, across various buildings, and contrast our performance against multiple state-of-the-art approaches.

9.3. RELATED WORK

FL was first introduced by Google in 2016 to address privacy concerns and reduce the communication costs associated with training ML models across decentralized devices [167]. By allowing devices to collaboratively learn from local data without sharing it, FL enhances privacy and reduces bandwidth requirements. This paradigm shift has gained traction in various fields, particularly in indoor localization systems, where FL has become popular for its ability to reduce the data collection efforts, preserve privacy, and adapt to variations in RSS due to device heterogeneity and environmental conditions [177].

Traditional non-FL-based solutions like DNNLOC [121], VITAL [80], TIPS [39], STELLAR [134], and ANVIL [45] have shown potential for indoor localization, particularly in addressing variations due to device heterogeneity. However, these solutions often require large-scale data collection to train the ML models effectively. To alleviate this burden, frameworks such as SANGRIA [82], DATALOC [178], and WiDeep [99] employ data augmentation techniques to improve model performance in heterogeneous environments. Despite these efforts, these models are static and cannot adapt in real-time to environmental changes, limiting their effectiveness in

practical indoor settings. In contrast, FL-based approaches can better adapt to environmental changes and can also minimize data collection overheads.

One of the early FL-based solutions in indoor localization was KRUM [179], which employed a simple Multi-Layer Perceptron (MLP) as the GM and used Euclidean distance-based filtering to select the LM update that deviated the least from the majority. While KRUM [179] reduces data collection overheads, it fails to incorporate collaborative learning from all clients, limiting its resilience to device heterogeneity effects. FEDLOC [175] improved upon KRUM by adopting a deep neural network (DNN) as the GM and utilizing FEDAvg for LM weight aggregation. While FEDAvg allows for collaborative learning and improves heterogeneity resilience, it introduces biases from noisy LMs, which can degrade GM performance. To address this, FEDHIL [88] introduced a DNN-based GM with a domain-specific aggregation strategy that selectively incorporated relevant weight tensors from LMs. This approach improves the robustness of the GM by reducing the impact of noisy or erroneous updates. However, despite these advancements, all of these methods remain vulnerable to data poisoning attacks, where malicious clients alter their local RSS data to produce erroneous LM updates that, when aggregated, corrupt the GM.

While no prior work has yet proposed a solution for addressing data poisoning attacks for indoor localization solutions, there have been some efforts for data poisoning attack mitigation in general FL systems. FEDCC [180] addresses data poisoning in FL systems by employing clustering techniques to group LMs based on gradient similarity, allowing it to detect and exclude poisoned updates from the GM aggregation. However, this approach may inadvertently filter out legitimate updates, particularly in heterogeneous environments. To further refine defense mechanisms, FEDLS [181] employs autoencoder-based latent space representations to detect anomalous LM updates. While this method enhances the detection of subtle poisoning attacks, it

is resource-intensive, limiting its deployment on mobile devices. Similarly, ONLAD [182], another anomaly detection framework, uses a semi-supervised autoencoder to detect poisoning on mobile devices. However, both FEDLS and ONLAD face challenges due to their high resource demands, as they employ two separate ML models for poison detection and localization. Additionally, both frameworks rely on FEDAvg for aggregation, which still introduces biases from noisy LMs into the GM.

From the analysis of prior works, it is evident that existing solutions either fail to address the challenges of device heterogeneity and data poisoning, or introduce significant computational overhead, particularly in the presence of data poisoning attacks. To the best of our knowledge, our proposed FL-based framework (SAFELOC) is the first to address data poisoning attacks in the indoor localization domain. The framework also co-addresses uncertainties arising from heterogeneous client mobile devices. A fused neural network architecture is introduced to perform poison detection, poison de-noising, and localization within a single model, significantly reducing computational overhead without compromising accuracy. Additionally, a novel saliency map-based aggregation technique helps invert the effects of poisoned weight tensors, enhancing localization performance further, across diverse real-world scenarios.

9.4. DATA POISONING ATTACKS IN FL INDOOR LOCALIZATION

While FL systems offer significant privacy and collaboration benefits, they are susceptible to various forms of data poisoning attacks. These attacks are particularly harmful in FL-based indoor localization frameworks, as they allow malicious clients to deliberately manipulate their LMs and degrade the overall performance of the GM, which leads to inaccurate location predictions.

In a typical FL-based indoor localization framework, as depicted in Figure 82, a central server maintains a GM that is distributed to several clients (mobile devices). Each client collects local RSS data from nearby APs and uses it to predict its location. The predicted label and local RSS data are then used to re-train the GM copy on the device, generating an LM. These LMs are uploaded back to the server, where an aggregation phase updates the GM based on the received LMs. This iterative process continues to improve localization performance as more local data is incorporated into the GM. However, among all the clients, there may be malicious clients who submit poisoned LM updates to the server. These poisoned updates, if integrated into the GM, can degrade the entire system's accuracy.

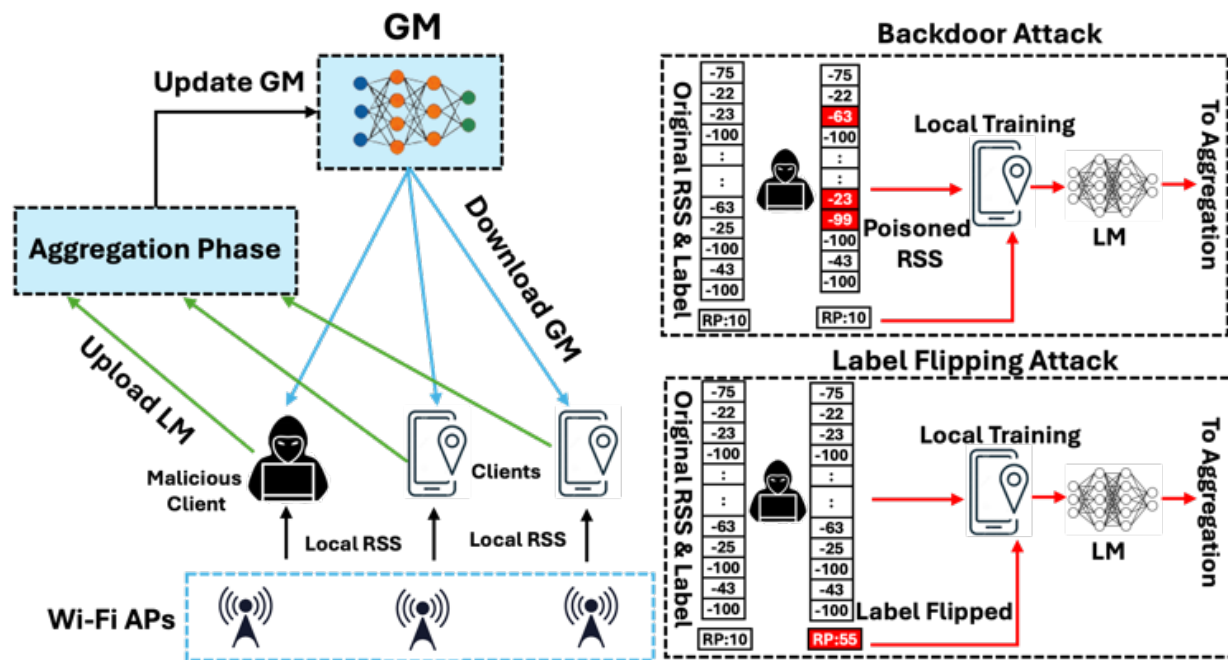


Figure 82: Working of an FL-based indoor localization framework with malicious clients (with backdoor and label-flipping attacks).

Data poisoning attacks in FL can generally be classified into two types: backdoor attacks and label-flipping attacks. While both target the integrity of the LMs before they are uploaded to the server, they differ in their execution and overall impact:

- **Backdoor Attacks:** Here, a malicious client deliberately poisons its local RSS fingerprints to train a poisoned LM. This LM, when uploaded to the server, influences the GM to behave incorrectly for specific inputs.
- **Label-Flipping Attacks:** Here, instead of modifying RSS fingerprints, the attacker retains correct RSS data but intentionally flips labels associated with the location data. As a result, LM updates are trained to associate valid RSS fingerprints with the wrong locations.

9.5. DATA POISONING ATTACK METHODS

We consider four different methods to generate backdoor attacks within FL-based indoor localization systems: Clean Label Backdoor (CLB), Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), and Momentum Iterative Method (MIM). Additionally, we also consider label-flipping attacks. These five methods aim to degrade localization accuracy by altering RSS fingerprints or labels during local training. These techniques are described below:

- **Clean Label Backdoor (CLB):** The CLB attack inserts perturbations into the LM by modifying the RSS fingerprints while leaving the label unchanged. The attack calculates a mask value along with the perturbation strength to generate poisoned samples. The CLB attack can be described as follows:

$$X_{CLB} = X + \epsilon * \delta(\nabla J(X, Y)) \quad (45)$$

where, X is the local RSS fingerprint, ϵ is the perturbation magnitude, and δ is the mask value that is computed using the gradients of the GM's loss function $\nabla J(X, Y)$. This generates a new RSS fingerprint X_{CLB} , designed to poison the GM through aggregation.

- **Fast Gradient Sign Method (FGSM):** The FGSM attack is a one-step, non-iterative attack method that calculates perturbations using the gradient of the GM's loss function with respect to the local RSS data. The FGSM attack is implemented using the equation below:

$$X_{FGSM} = X + \epsilon * \text{sign}(\nabla J(X, Y)) \quad (46)$$

where ϵ is the perturbation magnitude, and $\nabla J(X, Y)$ represents the gradient of the GM's loss function with respect to the local RSS fingerprint X . This perturbation generates a new RSS fingerprint X_{FGSM} , designed to poison the GM through aggregation.

- **Projected Gradient Descent (PGD):** The PGD attack is an iterative method that generates perturbations by applying small updates to the local RSS based on the GM's gradient. It is an iterative version of FGSM and ensures that the perturbation remains within a predefined bound. The PGD attack is given by:

$$X_{PGD} = X + Proj_{X,\epsilon} \left\{ \epsilon * \frac{\nabla J(X,Y)}{L|\nabla J(X,Y)|_2} \right\} \quad (47)$$

where $Proj_{X,\epsilon}$ ensures that the induced perturbation remains within a certain bound (X, ϵ) , $\nabla J(\theta, X, Y)$ represents the gradient of the GM's loss function and $L|\nabla J(\theta, X, Y)|_2$ represents the squared L2 norm (ridge regularization) of the gradients of the GM.

- **Momentum Iterative Method (MIM):** The MIM attack builds on PGD by incorporating momentum into the gradient updates, making the attack more effective by maintaining direction across iterations. The MIM attack can be described as follows:

$$X_{MIM} = \alpha * X + Proj_{X,\epsilon} \left\{ \epsilon * \frac{\nabla J(X,Y)}{L|\nabla J(X,Y)|_2} \right\} \quad (48)$$

where, α represents the momentum term. This method often leads to very potent data poisoning samples.

- **Label-Flipping:** The label-flipping attack leaves the local RSS data X unmodified but flips its corresponding label y to an incorrect $y_{FLIP} (FLIP)$, which leads the LM to associate valid RSS data with incorrect locations. The label-flipping attack can be described as follows:

$$y_{FLIP} = FLIP(y) \quad (49)$$

where $FLIP(y)$ denotes the operation that switches the original label y to an incorrect class

y_{FLIP} .

9.6. SAFELOC FRAMEWORK

The SAFELOC framework consists of two primary components: a fused neural network as the GM and a saliency map-based aggregation strategy to mitigate the effects of data poisoning attacks. The framework starts with training the fused neural network on a centralized server using a subset of RSS fingerprints from the building floorplan. This fused neural network integrates an autoencoder for poison detection and poison de-noising, alongside a classification layer for location prediction. By sharing layers between the autoencoder and classification layers, computational overhead is reduced, making the model suitable for resource-limited mobile device deployment. During training, the autoencoder encodes the RSS fingerprints into a low-dimensional latent space and reconstructs them using the de-noising decoder. The reconstruction error (RCE) is calculated as the difference between the original and reconstructed RSS data. This error is used to establish a threshold (τ) for detecting poisoned data during client-side training. Simultaneously, the classification layer learns to map RSS fingerprints to corresponding location labels. After training, the GM is distributed to clients, who use it to evaluate their local RSS data.

The modified autoencoder first checks for backdoor poisoning by comparing the RCE of the local RSS fingerprints to τ set during initial training. If the $RCE \leq \tau$, the data is considered clean, and the classification layer generates location predictions. The client then retrain its LM on this data and uploads it to the server for aggregation, to globally improve indoor localization accuracy for all clients. If the $RCE > \tau$, the RSS data is flagged as potentially poisoned via a backdoor attack. The reconstructed data, obtained by the de-noising decoder, is passed through the classification

layer to generate location coordinates. In this case, the de-noising decoder removes (by de-noising) the backdoor poisoned perturbations, and the LM is uploaded to the server.

In the case of label-flipping attacks, the attacker flips the predicted location coordinates before updating the LM, causing the wrong class to be updated even though the RSS fingerprint remains clean. When this poisoned LM is sent to the server, it can cause LM weight tensors to be significantly different from that of the GM. To counter this, a saliency map-based aggregation strategy is designed to compare the LM's weight tensors with those of the GM. A saliency map matrix is created where similar tensors are assigned high saliency values, and highly deviated tensors are assigned low values. The saliency map is then used to adjust the LM weight tensors before performing aggregation with the GM.

In the following subsections, we describe the working of the fused neural network and saliency map-based aggregation strategy used in the SAFELOC framework.

9.6.1. FUSED NEURAL NETWORK ARCHITECTURE

The fused neural network in SAFELOC integrates an autoencoder for poison detection and de-noising alongside a classification layer for location prediction, as seen in Figure 83. The autoencoder is composed of an encoder and a de-noising decoder. The encoder compresses the RSS fingerprint data into a progressively smaller latent space using fully connected layers, reducing the number of neurons at each layer until the bottleneck layer. This bottleneck represents a low-dimensional latent space that captures the key features of the input data. The latent space is then bifurcated into two outputs: one directed to the de-noising decoder for reconstruction and the other to the classification layer for location prediction.

The de-noising decoder uses fully connected layers with an increasing number of neurons, structured similarly to the encoder's architecture but in reverse, to reconstruct the original RSS fingerprint. To improve precision of reconstruction, we freeze the gradients from the encoder and propagate them to their corresponding layers in the decoder. This approach helps maintain the learned patterns during the encoding process while ensuring the decoder properly removes any perturbations, particularly from backdoor poisoning attacks. The RCE is calculated using the Mean Squared Error (MSE) between the input RSS fingerprint and the reconstructed RSS fingerprint. If $RCE \leq \tau$, the latent space representation is forwarded to the classification layer for location prediction. If the $RCE > \tau$, the reconstructed fingerprint undergoes another encoding step (re-supplied to the encoder as input) to generate a new latent space, which is then passed to the classification layer for location predictions. This approach mitigates backdoor poisoning by removing perturbations during the reconstruction phase. When label-flipping attacks occur, the input RSS data remains clean, but the label is manipulated before the LM update. This situation is addressed in the server through the saliency map-based aggregation method, which adjusts LM weight tensors to safeguard the GM from poisoning, as discussed next.

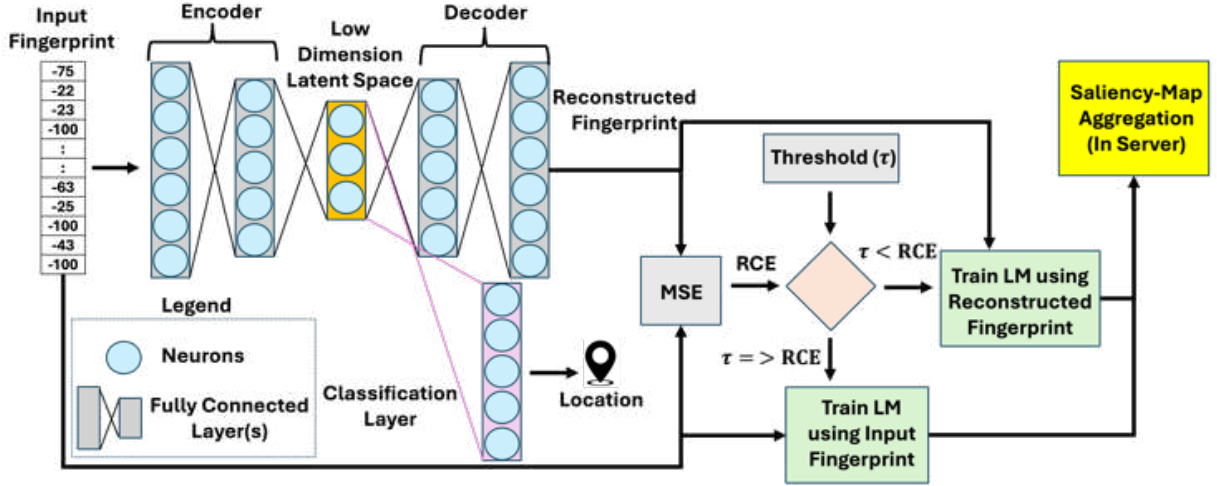


Figure 83: Fused neural network architecture used in SAFELOC.

9.6.2. SALIENCY-MAP BASED AGGREGATION

The saliency map-based aggregation strategy is used to mitigate the effects of poisoned LMs by adjusting their weight tensors before aggregation. Once the server receives the LMs, it first computes a deviation matrix (ΔW_i) to quantify the difference between the weight tensors of the LM and the GM, which is calculated as:

$$\Delta W_i = |W_{LM,i} - W_{GM,i}| \quad (50)$$

where i represents the index of the individual weight tensors, $W_{LM,i}$ represents the weight tensors of the LM, and $W_{GM,i}$ represents the weight tensors of the GM. To counter the influence of these deviations, we calculate a saliency matrix (S_i) for each weight tensor using the inverse deviation method, as shown in the equation:

$$S_i = \frac{1}{1 + \Delta W_i} \quad (51)$$

The S_i adjusts the magnitude of each weight tensor in $W_{LM,i}$ before aggregation. It assigns higher saliency values to weight tensors that closely match $W_{GM,i}$ (i.e., small deviation) and lower saliency values to weight tensors that show significant deviation. This ensures that large deviations are given less weight during aggregation. Once the S_i is computed, it is used to adjust the $W_{LM,i}$ before aggregation. The adjusted LM weight tensors (W_{Adj}) are obtained by multiplying the $W_{LM,i}$ with the S_i values, as shown below:

$$W_{Adj} = S_i * W_{LM,i} \quad (52)$$

This results in an adjusted LM that accounts for deviations while mitigating the potential influence of poisoned weights. The W_{Adj} is then aggregated with the GM using:

$$W'_{GM} = W_{GM} + W_{Adj} \quad (53)$$

where W'_{GM} represents the updated GM. This process reduces the impact of poison updates on the GM, ensuring that the GM remains robust against poisoned LM updates.

9.7. EXPERIMENTAL SETUP

In our experiments, we evaluate the performance of the SAFELOC framework across 1) real-world RSS fingerprint data from different buildings, 2) various data poisoning attack scenarios,

and 3) contrast performance with state-of-the-art frameworks, including FEDHIL [88], FEDLOC [175], FEDCC [180], FEDLS [181], and ONLAD [182].

We collected RSS fingerprint data from six different client mobile devices (Samsung Galaxy S7, OnePlus 3, Motorola Z2, LG V20, BLU Vivo 8, HTC U11). Data was collected along paths from five different buildings, with each path having different numbers of reference points (RPs) and different numbers of visible APs. Building 1 comprises 60 RPs with up to 203 visible APs, Building 2 consists of 48 RPs with 201 visible APs, Building 3 includes 70 RPs with 187 visible APs, Building 4 has 80 RPs with 135 visible APs, and Building 5 features 90 RPs with 78 visible APs. The granularity for all RPs is set at 1 meter, which we consider sufficient for indoor localization. We standardized the RSS values between 0 dBm (strongest signal) and -100 dBm (weakest signal). Training was done on data collected from one device (Motorola Z2), and testing on the remaining five devices. Training involved collecting five fingerprints per RP per building, while testing used one fingerprint per RP per building. This method strikes a balance between gathering sufficient training data and the practicality of data collection across real-world environments.

The SAFELOC framework was configured with specific hyperparameters: the encoder in the fused neural network consists of three fully connected layers with 128, 89, and 62 neurons, while the decoder has two fully connected layers with 89 and 128 neurons. ReLU activation was applied to all layers, and the autoencoder was trained using the MSE loss function. The classification layer consists of neurons corresponding to the number of RPs for each building. Sparse categorical cross-entropy was set as the classification layer's loss function. Both the autoencoder and the classification layer were trained using the Adam optimizer with a learning rate of 0.001 for 700

epochs. For client-side training, a reduced learning rate of 0.0001 and 5 epochs were used to facilitate lightweight local training.

In the following subsections, we present results of experiments to determine the optimal reconstruction threshold (τ), examine SAFELOC's performance under various data poisoning attacks and attack intensities, and compare SAFELOC's localization accuracy with current state-of-the-art frameworks. We also present data on SAFELOC's model inference latency and parameter footprint, and an analysis of the framework's scalability.

9.7.1. DETERMINATION OF RECONSTRUCTION THRESHOLD (τ)

As a first step, we analyzed the effect of varying the reconstruction threshold (τ) on localization performance in SAFELOC. τ plays a crucial role in determining whether local RSS fingerprints collected by clients contain backdoor poisoning, based on the reconstruction error (RCE). We vary τ from 0 to 0.5 to find the optimal threshold that minimizes the mean localization error across the five buildings, calculated by averaging results from all six devices and RPs. This threshold allows variance for device heterogeneity and identify the most effective τ value for poison detection. The HTC U11 device is used to introduce data poisoning attacks (CLB, FGSM, PGD, MIM, and label flipping), with perturbation strength (ϵ) varying from 0 to 1 in steps of 0.01 up to 0.1, then in steps of 0.1. Figure 84 shows that as τ increases from 0.05 (5% tolerance) to 0.5 (50% tolerance), localization errors vary. The lowest mean error is achieved at $\tau = 0.1$, particularly for Building 4, where error variance is minimal. Localization errors remain stable between $\tau = 0.15$ and 0.25, but beyond $\tau = 0.3$, there is a significant increase in errors, peaking between $\tau = 0.45$ and 0.5. This indicates that higher τ values allow more poisoned RSS data to influence the GM, leading to larger errors. Based on these observations, we determine that $\tau = 0.1$, allowing a 10% variance, is optimal

for minimizing localization errors while effectively detecting poisoned data. This τ will be used in subsequent evaluations under different poisoning scenarios.

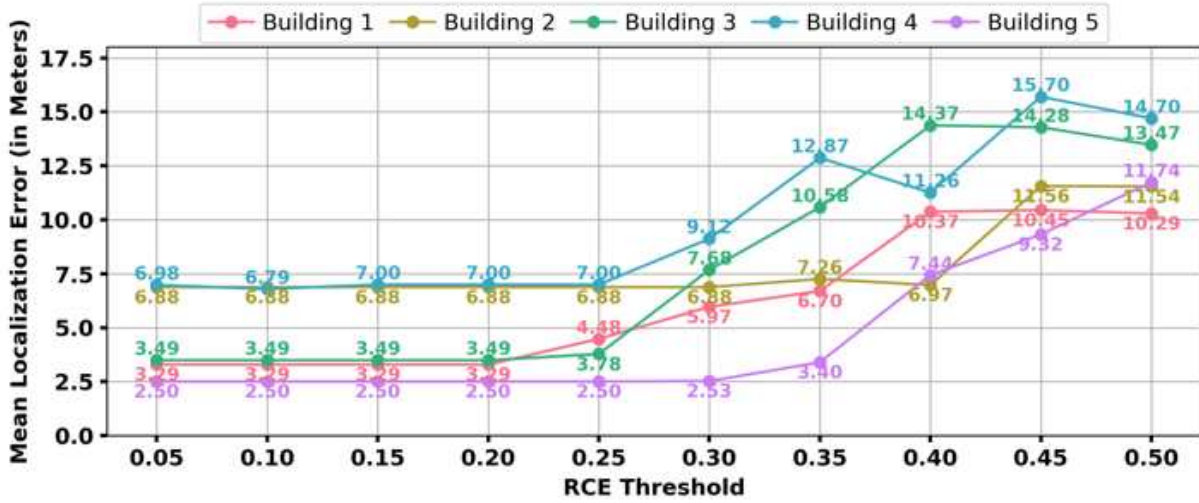


Figure 84: Impact of varying the reconstruction threshold (τ) on mean localization error across five building floorplans.

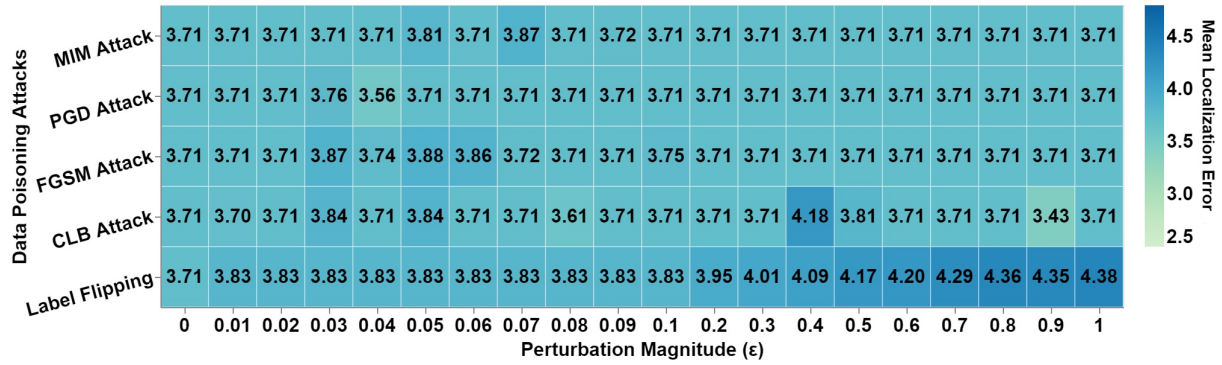


Figure 85: Mean localization error under various data poisoning attacks and perturbation magnitudes (ϵ).

9.7.2. EFFECTS OF DATA POISONING ATTACKS IN SAFELOC

Next, we investigate the impact of various data poisoning attacks on the SAFELOC framework, focusing on both low ($\epsilon = 0.01$ to 0.09 , indicating 1% to 9%) and high ($\epsilon = 0.1$ to 1.0 ,

indicating 10% to 100%) perturbation strengths, as depicted in Figure 85. Each row in the heatmap represents a different data poisoning attack, and each column represents a different ϵ value. The HTC U11 device is again used to introduce data poisoning attacks. The corresponding mean localization error across all devices, buildings, and RPs is recorded in each cell in the Figure. It can be observed that up to $\epsilon < 0.1$, SAFELOC maintains a stable mean localization error across all attacks, even as perturbation magnitude increases. This stability indicates that the autoencoder and saliency map-based aggregation can effectively handle small perturbations. As we shift to $\epsilon > 0.1$, the mean errors remain stable for backdoor poisoning attacks, but for label-flipping attacks, we observe a slight increase in mean errors beginning at $\epsilon = 0.2$, reaching up to 4.38 meters at $\epsilon = 1.0$. The saliency map method, although highly effective, may assign relatively higher saliency to these poisoned weight tensors, leading to a slight error increase. Despite the small rise in mean errors at higher ϵ values the SAFELOC consistently outperforms state-of-the-art frameworks, as discussed next.

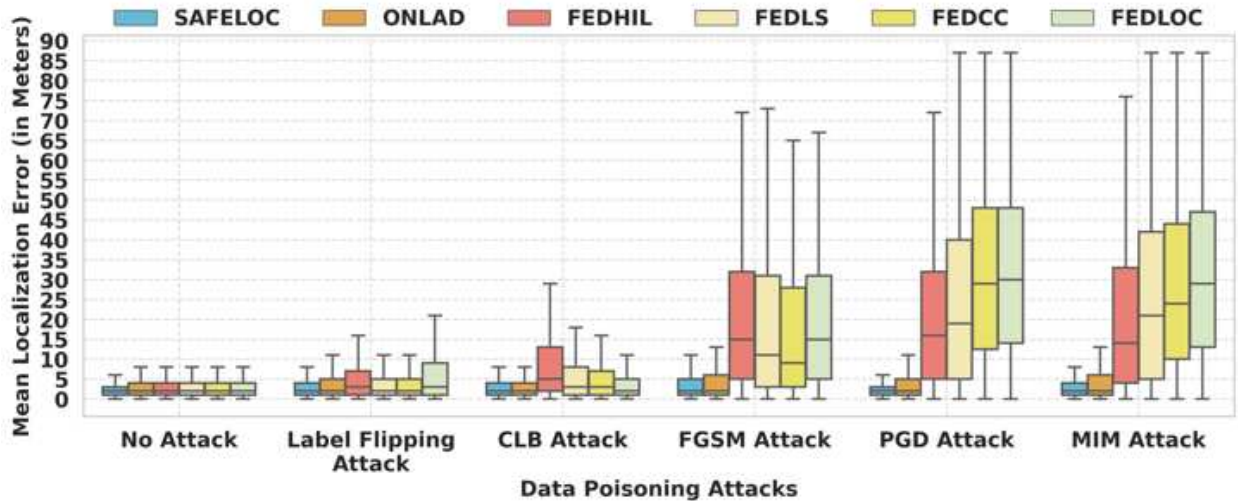


Figure 86: Comparison of SAFELOC against state-of-the-art.

9.7.3. COMPARISION WITH STATE-OF-THE-ART

We compared the performance of the SAFELOC framework against several state-of-the-art indoor localization frameworks. The box and whisker plot in Figure 86 shows worst-case errors (upper whisker), best-case errors (lower whisker), and mean errors (center bar) across various poisoning scenarios. The results are aggregated across all buildings.

We observe that SAFELOC consistently achieves lower mean and worst-case localization errors compared to all the other frameworks. ONLAD ranks second due to its distinct poison detection and localization architecture, but its use of FEDAvg allows noisy weight tensors to degrade the GM, particularly for label flipping, FGSM, and MIM attacks. FEDLS performs well under label flipping attacks but suffers in backdoor attacks as its autoencoder latent space is vulnerable to stronger perturbations. FEDCC, while resilient to label flipping, is weak against backdoor attacks, as it inadvertently filters legitimate LMs under severe perturbations. FEDHIL's selective weight aggregation aggregates large tensor changes caused by attacks, leading to higher errors, particularly in backdoor scenarios.

Lastly, FEDLOC exhibits the highest localization errors across all attack types due to its lack of defense mechanisms. SAFELOC achieves $1.2\times$ to $2.11\times$ lower mean errors and $1.16\times$ to $2.33\times$ lower worst-case errors for label flipping, and $1.33\times$ to $5.9\times$ lower mean errors and $1.5\times$ to $7.8\times$ lower worst-case errors for backdoor attacks (CLB, FGSM, PGD, MIM) compared to other frameworks. Table 17 shows the implementation overheads of all compared frameworks. SAFELOC has the least model parameters and the lowest model inference latency, offering $1.04\times$ to $2.1\times$ faster inference than the state-of-the-art frameworks.

Table 17 : model latency and parameters comparison

Framework	Model Inference Latency	Total Parameters
SAFELOC	64 ms	41,094
ONLAD	87 ms	130,185
FEDHIL	84 ms	97,341
FEDCC	67 ms	42,993
FEDLS	103 ms	282,676
FEDLOC	135 ms	137,801

9.7.4. SCALABILITY ANALYSIS WITH INCREASING NUMBER OF CLIENTS

Lastly, we evaluate SAFELOC’s scalability against the two best frameworks from prior works (ONLAD, FEDHIL) by increasing clients from 6 to 24, with poisoned clients rising from 1 to 12. Figure 87 shows FEDHIL exhibits a steady rise in mean localization errors as more clients are introduced, particularly with an increase in poisoned clients, while ONLAD and SAFELOC maintain stable performance. SAFELOC consistently shows the lowest mean localization errors, owing to its saliency map aggregation that mitigates the impact of poisoned clients, even as their number grows.

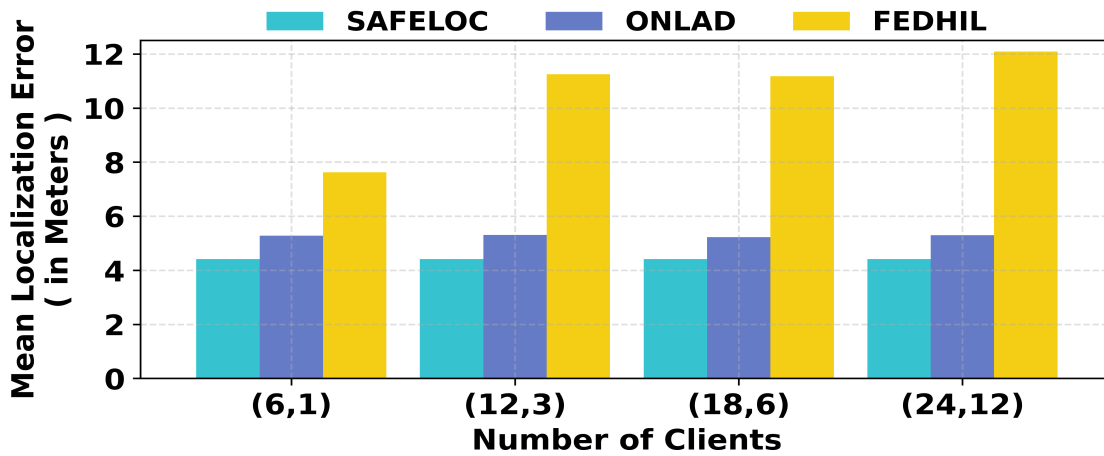


Figure 87: Mean localization error with increasing number of (total, poisoned) clients ratio.

9.8. CONCLUSIONS

In this chapter, we introduced SAFELOC, a novel federated learning-based indoor localization framework designed to address the challenges of device heterogeneity and data poisoning attacks. By combining a fused neural network for integrated data poisoning detection, de-noising, and localization with a dynamic saliency map-based aggregation strategy, SAFELOC effectively mitigates malicious client updates while maintaining a lightweight model suitable for mobile deployment. Experimental results demonstrate that SAFELOC achieves up to a $5.9\times$ reduction in mean localization error, a $7.8\times$ decrease in worst-case error, and a $2.1\times$ faster inference latency compared to state-of-the-art frameworks.

CHAPTER 10

CIELO: CLASS-INCREMENTAL CONTINUAL LEARNING FOR OVERCOMING CATASTROPHIC FORGETTING WITH SMARTPHONE-BASED INDOOR LOCALIZATION

With dynamically evolving indoor environments, class-incremental learning (CIL) is important for smartphone-based indoor localization systems to adapt to new indoor areas. However, CIL has many associated challenges such as catastrophic forgetting and high data storage demands when new classes are introduced. To address these critical challenges, we present CIELO, a novel framework that utilizes an innovative representation memory management approach with crowdsourcing to enable high accuracy CIL while reducing data storage requirements. Experimental results across multiple paths and devices indicate that CIELO improves indoor localization accuracy by up to $29.4\times$ with new classes, reduces data storage by up to $1.75\times$, and lowers edge server device power consumption by up to $1.69\times$ compared to the state-of-the-art.

10.1. INTRODUCTION

Indoor localization systems are being adopted across a wide range of applications, including augmented and virtual reality (AR/VR), smart home automation, asset tracking, and indoor navigation [170]. This widespread adoption is fueling the rapid growth of the indoor localization market, which is projected to reach USD 83.1 billion by 2030, as demand rises for precise positioning solutions across diverse sectors [185].

To aid with indoor localization across environments where GNSS/GPS signals are absent (e.g., buildings, urban areas, underground mines), industries are exploring a range of wireless RF

technologies (e.g., Wi-Fi, Bluetooth, and Ultra-Wideband (UWB)) [186], [187], [188], channel characteristics (e.g., channel state information (CSI), received signal strength (RSS) [189]) and localization techniques (e.g., trilateration, dead reckoning, fingerprinting [135]). Out of these, Wi-Fi RSS-based fingerprinting has emerged as a prominent choice for smartphone-based indoor localization for users [90]. This can be largely attributed to the widespread availability of Wi-Fi in indoor locales and its seamless integration with modern mobile devices, as well as the relatively higher accuracy of RSS fingerprinting [190]. RSS fingerprinting involves capturing signal strength values from multiple Wi-Fi access points (APs) to create unique "fingerprints" associated with specific locations or reference points (RPs) within an indoor environment. These fingerprints are then trained using machine learning (ML) models to predict a device's location by learning patterns in the RSS data, making the approach suitable for real-time deployment in smartphones.

Wi-Fi RSS fingerprinting-based indoor localization systems typically operate in two phases: offline and online [135]. During the offline phase, RSS fingerprints are collected at various RPs, and an ML model is trained to learn location-specific patterns. In the online phase, the trained ML model is deployed on mobile devices, where it predicts the device's location in real-time using RSS values collected at unknown RPs. Despite the theoretical simplicity and effectiveness of this approach, indoor environments are inherently dynamic, which introduces noise into fingerprints, i.e., RSS fingerprint measurements are prone to variations caused by factors such as obstacles, human movement, and environmental phenomena such as multipath fading effects. Such variations degrade the performance of static ML models, which struggle to adapt to continuous changes in the environment, limiting their accuracy in dynamic scenarios.

Moreover, extending a deployed ML model to localize new paths (those not included in offline training) or to adapt to layout changes presents several challenges. The model relies on pre-stored

RSS fingerprints for training, and incorporating new path data requires: 1) expanding this fingerprint database to accommodate new RP data, which increases storage requirements, and 2) retraining the ML model which adds computational and time overhead. Adapting to the vagaries of dynamic environments requires continuous updates to the fingerprint database and ML model, a process that remains an open challenge for robust and efficient indoor localization.

Continual learning (CL) represents a promising solution for this problem, offering the capability to update an ML model dynamically with new data while retaining previously learned knowledge [191]. Unlike conventional ML models, which are designed to work with static data distributions, CL focuses on adapting to dynamic and evolving data distributions. By integrating crowdsourcing into this process, new indoor localization data can be gathered from mobile users over time. This data contributes to updating a centralized ML model, referred to as the global model (GM), which enables the system to continuously learn and adapt to new RPs or locations not covered during the initial offline training phase, significantly enhancing its applicability to evolving paths and new environments.

CL scenarios are often categorized based on how the input and output distributions evolve. For instance, some approaches adapt to changes in input data distributions without expanding the outputs (not introducing new RPs), while others require the system to handle entirely new outputs, such as additional RPs [192]. Among these, Class-Incremental Learning (CIL) is considered the most challenging [193], as it requires the GM to predict an ever-growing set of outputs, such as new RPs, without explicitly knowing which part of the data belongs to previous or new distributions. Implementing CIL for indoor localization introduces two major challenges: catastrophic forgetting and data storage overheads. Catastrophic forgetting arises when continual updates to the GM result in the loss of knowledge from previously learned RPs [194], [195], [197].

As new RPs are introduced, the GM may become biased towards these new locations, effectively "forgetting" earlier learned RPs and causing a decline in localization accuracy. A naive solution would involve storing data for all previously learned RPs and periodically retraining the GM on all of this data. However, this approach is impractical due to the high storage and computational costs associated with it.

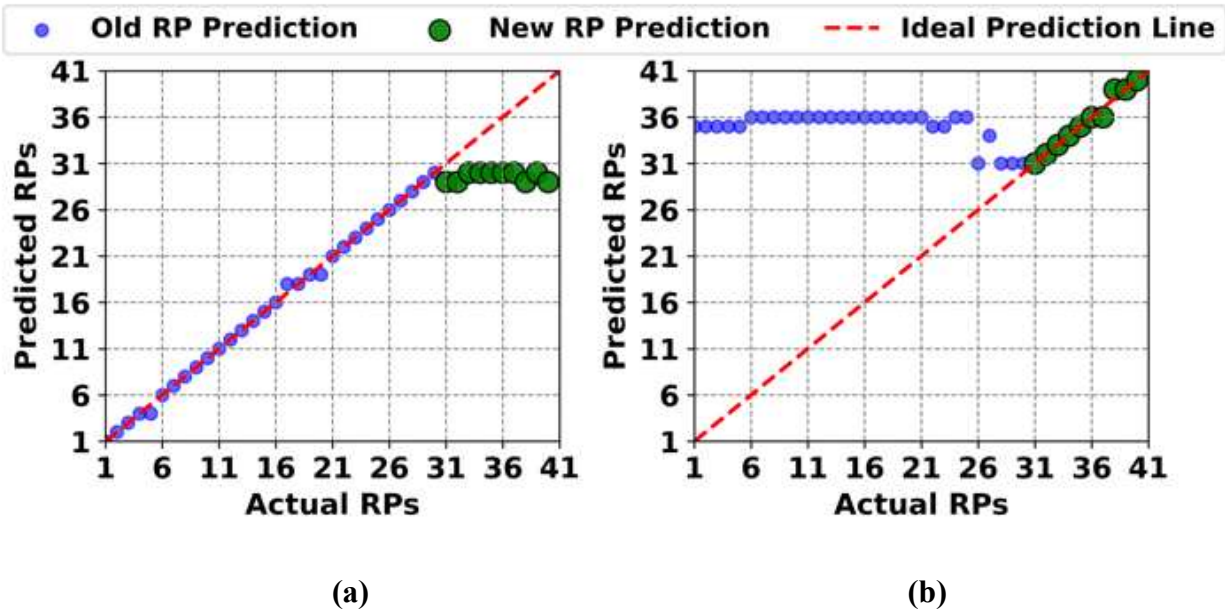


Figure 88: Illustration of catastrophic forgetting using a four-layer DNN. (a) DNN predictions after training on initial 30 RPs (0 to 30). (b) DNN predictions after retraining exclusively on 10 new RPs (31 to 40).

To illustrate the impact of catastrophic forgetting, we conducted an experiment using a four-layer deep neural network (DNN) as the GM for Wi-Fi RSS fingerprinting-based indoor localization in a real building. Initially, the DNN was trained offline on RSS fingerprints from 30 RPs (0 to 30). Subsequently, it was updated with data from 10 additional RPs (31 to 40) crowdsourced from users' smartphones. Figure 88 presents the results of this experiment. In Figure 88 (a), the DNN is trained exclusively on the initial 30 RPs and it performed well when tested on

these same RPs, demonstrating accurate classification. However, the DNN failed to accurately classify the new RPs, misinterpreting them as existing classes, as the new RPs were not a part of the initial training data. In Figure 88 (b), the DNN was retrained exclusively with data from the 10 new RPs, resulting in improved classification accuracy only for the new RPs but a marked decline in performance for the original 30 RPs. This outcome suggests that the DNN replaced its prior knowledge of the initial 30 RPs with information about the newly added 10 RPs, a clear demonstration of catastrophic forgetting. These results highlight the pressing need for a method that can effectively integrate new RPs while preserving knowledge of previously learned RPs, without retraining the GM on the entire dataset repeatedly.

10.2. CONTRIBUTIONS

To address these important challenges, we propose CIELO (Class-Incremental Learning for Indoor Localization), which to the best of our knowledge, is the first framework to introduce CL for smartphone-based indoor localization. CIELO is designed to address the unique challenges of this domain, incorporating an innovative CIL approach to continually update the GM with new RPs while retaining prior knowledge, with low data storage overheads. Our key contributions in this work are:

- We propose a novel CIL framework that incrementally learns new RPs while preserving accuracy on previously learned RPs, enabling the GM to learn new indoor paths.
- We present a novel representation memory management (RMM) policy to mitigate catastrophic forgetting while capping data storage requirements.
- We develop and deploy CIELO with optimizations for resource-limited smartphone devices and low edge server overheads.

- We conduct comprehensive real-world evaluations using data collected from multiple smartphones and indoor environments, benchmarking CIELO against state-of-the-art methods.

10.3. RELATED WORK

Wi-Fi fingerprinting-based indoor localization has gained significant attention, with research conferences like IEEE IPIN [199] and competitions hosted by companies such as Microsoft [34], driving innovation and advancing the field. Traditional ML methods such as KNN [201], HMM [48], and GPC [202] effectively address RSS fluctuations caused by factors such as human interference and signal shadowing [203]. Building on these, deep learning frameworks like DNNLOC [121], CNNLOC [28], and ANVIL [45] leverage neural networks to improve feature extraction and localization accuracy. However, these models are inherently static and lack the ability to dynamically adapt to environmental changes. These advancements demonstrate progress but highlight the need for more adaptive solutions in dynamic environments.

Federated learning (FL) has emerged as an effective solution to address the limitations of traditional indoor localization solutions, by learning to aggregate and adapt to noisy information contributed by users. Early FL frameworks like KRUM [204] employed Multi-Layer Perceptrons (MLPs) and Euclidean distance-based filtering to enable localization. FEDLOC [175] improved upon KRUM by adopting a DNN as the GM and using the popular FedAvg (a technique to update the GM), enhancing resilience to RSS fluctuations but also introducing biases from noisy local data updates from users. FEDHIL [88] further refined this approach with domain-specific aggregation to reduce the impact of noisy updates, improving robustness. However, FEDHIL, like other FL frameworks, remains limited in its ability to handle new RPs or classes incrementally. Its

inability to learn new classes without retraining on the entire dataset constrain its applicability in dynamic indoor localization scenarios.

CIL is one of the most challenging scenarios in CL, requiring a GM to integrate new classes into an expanding output space. This setting is widely studied in domains like object recognition and language modeling, where it is critical for models to classify new categories without prior knowledge of task boundaries [194], [195]. A major challenge in CIL is catastrophic forgetting. To address this, several approaches have been explored in domains outside of indoor localization. Elastic Weight Consolidation (EWC) [196] constrains updates to GM weights critical for earlier tasks, helping mitigate forgetting. However, EWC requires computationally intensive weight importance matrices, making it impractical for resource-limited devices. Progressive Neural Networks (PNNs) [205] create new branches for each task, isolating knowledge and preventing interference, but it results in significant memory and computational demands. Learning Without Forgetting (LwF) [198] preserves outputs from earlier tasks to enable incremental learning, but its reliance on consistent task-specific outputs often fails in dynamic and non-IID (non-independent and identically distributed) environments, conditions commonly encountered in Wi-Fi RSS fingerprinting. Other methods, such as Generative Replay (GR) [206], generate synthetic data to simulate previous tasks, reducing the need for storing large datasets. However, GR's reliance on computationally intensive generative models makes it challenging for resource-limited devices. Incremental Classifier and Representation Learning (iCaRL) [200] uses a similarity-based sample selection approach to retain a small subset of (representative) data, offering memory efficiency. But iCaRL struggles to handle evolving data distributions in dynamic environments.

In summary, the methods EWC [196], PNNs [205], LwF [198], GR [206], and iCaRL [200] represent diverse approaches that attempt to mitigate catastrophic forgetting to some extent;

however, it remains an open challenge, particularly in dynamic environments where new classes are introduced incrementally. Moreover, these approaches often rely on significant data storage or computationally intensive resources, limiting their applicability to indoor localization systems. To address these challenges, we present CIELO, the first Class-Incremental Learning (CIL) framework designed specifically for indoor localization. CIELO integrates CL principles with resource-efficient strategies, enabling it to handle evolving environments and new RPs while maintaining robust performance.

10.4. CONTINUAL LEARNING: OVERVIEW

CL enables a GM to adapt to new data, overcoming the limitations of conventional training methods that assume simultaneous access to the entire dataset [191], [207]. This makes it relevant in dynamic environments where data evolves over time. CL scenarios are typically categorized into three types [193] based on how input (x) and output (y) distributions evolve over time: 1) Domain-Incremental Learning (DIL): Here for $f(x) = y$, x evolves but y remains fixed [208], [209]. The GM adapts to changes in input distributions such as RSS fluctuations, but as y remains fixed, it does not support learning new classes; 2) Task-Incremental Learning (TIL): Here, $f_t(x) = y_t$, where both x and y evolve, and the task identity (t) is provided. This knowledge enables the GM to apply task-specific adaptations, such as using separate classifiers or specialized modules for each task [210]. However, the reliance on t limits its applicability in scenarios where such information is unavailable; 3) Class-Incremental Learning (CIL): Here for $f(x) = \{y_1, y_2, \dots, y_n\}$ both x and y evolve, but task identity t is not provided. The GM must handle an expanding output space (e.g., any number of new RPs) while retaining its ability to map x to y for previously learned RPs. This is the most challenging type of CL problem, as the GM must balance

integrating new knowledge while preserving older one, for arbitrarily large numbers of new RPs [211].

10.4.1. CIL IN INDOOR LOCALIZATION

In CIL for indoor localization, a GM designed for the problem must learn new classes incrementally and expand its capabilities as new classes become available. Unlike traditional ML models, which are trained on a fixed number of classes, CIL systems dynamically update the GM as new classes are introduced by users. In the context of Wi-Fi RSS fingerprinting-based indoor localization, adding a new class corresponds to introducing a new RP to the GM. Each RP is treated as a unique "class" in the classification task, and CIL allows the GM to incorporate these new RPs during the online phase.

During the offline phase, the GM is initially trained with RSS fingerprints collected from a predefined set of RPs (crowdsourced from users initially), denoted as C_{old} , on an edge server. This GM is distributed to the user's mobile devices to enable them to perform indoor localization. When new RPs (C_{new}) are introduced in the online phase (e.g., after crowdsourcing them from users traversing new locations), the GM's architecture must be updated to include the additional classes. This requires modifying the output layer of the GM to accommodate the expanded set of classes. The GM's output layer typically uses a *softmax* function, which converts RSS values (logits) into probabilities for each class. The logits (Z) for the C_{old} are calculated as shown in Equation 54:

$$Z_{old} = W_{old} * H + B_{old} \quad (54)$$

where Z_{old} is a vector containing the logits for C_{old} , W_{old} is the weight matrix of the output (classification) layer, H is the feature vector produced by the final hidden layer of the GM summarizing the RSS fingerprint input, and B_{old} is the bias vector corresponding to the C_{old} classes. The *softmax* function is then applied to convert Z_{old} into probabilities (Y_i) for class i , as shown in Equation 55:

$$Y_i = \frac{\exp(Z_i)}{\sum_{j=1}^{C_{old}} \exp(Z_j)}, \text{ For } i = \{1, 2 \dots C_{old}\} \quad (55)$$

When C_{new} is introduced, the GM's output layer must be expanded to include the additional classes. This involves updating the W_{old} and B_{old} of the output layer to accommodate C_{new} , as shown in Equation 56:

$$W_{old,new} = [W_{old}, W_{new}]; B_{old,new} = [B_{old}, B_{new}] \quad (56)$$

where W_{new} and B_{new} are the weight matrix and bias vector of C_{new} . $W_{old,new}$ is this updated output layer and the new logits (Z_{new}) are computed as shown in Equation 57:

$$Z_{new} = W_{old,new} * H + B_{old,new} \quad (57)$$

where Z_{new} now contains logits for both the previously learned classes and the newly added ones. This process enables the GM to classify RSS fingerprints from both the original RPs and the newly introduced RPs, thereby dynamically expanding its localization classes. However, simply

expanding the GM's output layer in this manner introduces challenges, such as catastrophic forgetting.

10.4.2. CATASTROPHIC FORGETTING

A significant challenge in CIL, especially in dynamic environments like indoor localization, is catastrophic forgetting. This challenge arises when new RPs are introduced, requiring the GM to update its representation layers (such as output layer) to capture the unique signal patterns of these new RPs. However, without access to historical data, these updates often overwrite previously learned representations. This leads to a shift in the GM's weights, favoring the new RPs at the expense of earlier ones. As a result, the GM exhibits bias toward newly added classes (new RPs), while "forgetting" previously learned classes. This bias compromises localization accuracy for prior locations, as shown earlier in Figure 88. The gradual loss of earlier knowledge as new classes are introduced further highlights the limitations of conventional CIL methods [212]. To address this challenge, it is essential to adopt strategies that preserve earlier knowledge while seamlessly integrating new data. These strategies should ensure consistent and accurate localization performance across all RPs, both old and new. To overcome this limitation and facilitate seamless learning of new classes, we propose a new CIL method, which is described next.

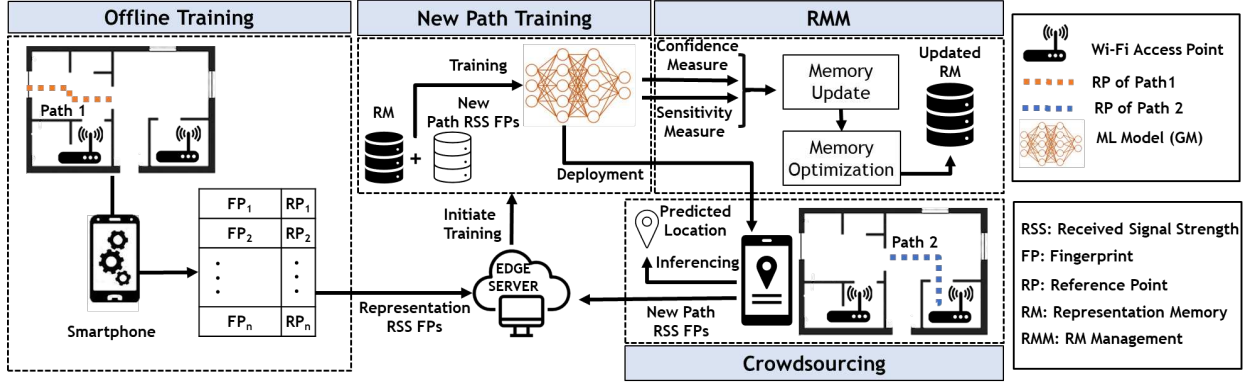


Figure 89: Overview of the CIELO framework for class incremental learning in Wi-Fi RSS fingerprinting-based indoor localization

10.5. CIELO FRAMEWORK

The CIELO framework begins with the offline phase, as illustrated in Figure 89. During this phase, RSS fingerprints are labeled for various RPs. The collected data is stored in the edge server as a representation memory (RM), which initially contains data for $\mathcal{C}_{old}$ (the original RPs). This $\mathcal{C}_{old}$ dataset is also used to train the GM during the offline phase. Once the GM is trained on $\mathcal{C}_{old}$, the trained GM is deployed on user devices (smartphones), where it is used to infer a user's location during the online phase. In the online phase, when a user encounters a new RP during the online phase, the system transitions into the CIL phase. At this stage, the user uploads the RSS fingerprints collected at the new RP ($\mathcal{C}_{new}$) to the edge server. The edge server processes this new data by combining it with samples from the existing RM ($\mathcal{C}_{old}$) to create a balanced training dataset that includes both old and new RPs ($\mathcal{C}_{old} + \mathcal{C}_{new}$). The GM is then retrained on this updated dataset to incorporate the new RPs. However, directly merging $\mathcal{C}_{old} + \mathcal{C}_{new}$ introduces several challenges: 1) It increases the risk of catastrophic forgetting, where the GM loses knowledge of previously learned RPs as it learns the new ones; 2) As more new RPs are added, the RM becomes bulky and large,

leading to higher storage requirements and increased GM retraining overheads. These challenges necessitate a strategic memory management policy to effectively manage the RM.

CIELO addresses these challenges with a novel representation memory management (RMM) policy, which operates via two key processes: Memory Update and Memory Optimization. These processes ensure that the RM retains only the most diverse and representative data samples, preserving critical features from previous RPs while minimizing storage and computational overheads. By doing so, CIELO enables CIL with minimal data storage (RMM footprint) while effectively mitigating the catastrophic forgetting challenge, making it highly suitable for dynamic indoor localization systems.

10.5.1. REPRESENTATION MEMORY MANAGEMENT (RMM)

The RMM policy is a critical component of the CIELO framework that ensures effective handling of data while mitigating the challenges of catastrophic forgetting and storage overhead.

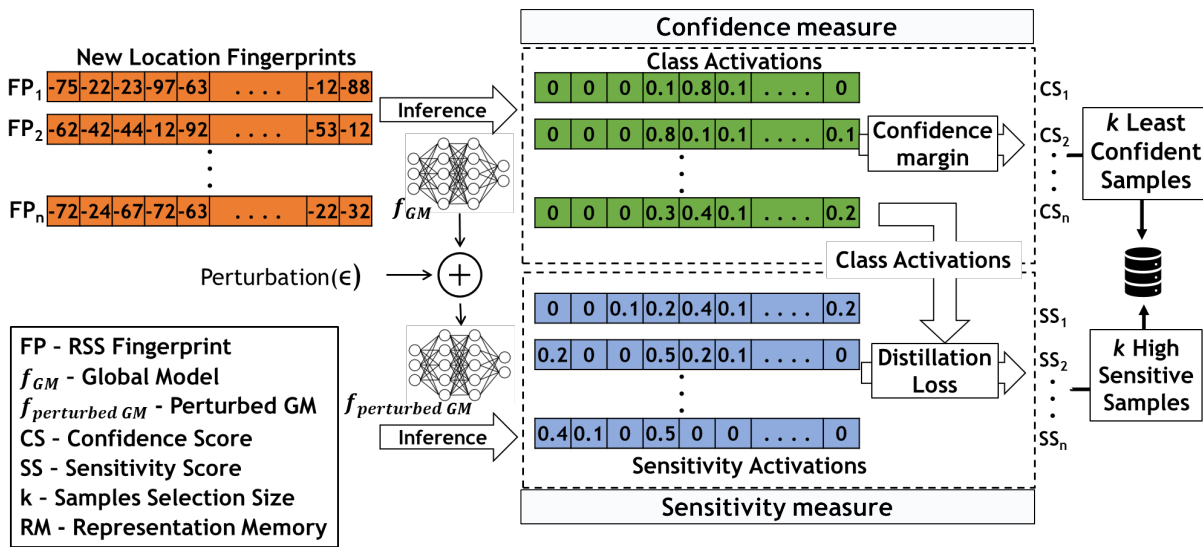


Figure 90: Confidence and sensitivity measure for memory management

As shown in Figure 90, RMM operates by evaluating and managing the RM through two main processes: memory update and memory optimization. These processes work in tandem to retain only the most diverse and representative data samples, enabling the GM to integrate new RPs seamlessly while preserving knowledge of previously learned ones. The RMM evaluates RSS fingerprints using two measures—Confidence Measure and Sensitivity Measure—to determine the importance of each fingerprint. This evaluation helps prioritize data samples for retention in the RM, ensuring that the memory remains compact while maintaining its relevance to both previously learned and newly added RPs.

- **Memory Update:** The memory update process is responsible for selecting new RSS fingerprint samples for inclusion in the RM. This selection is based on the importance of the samples, as determined by the confidence measure and sensitivity measure metrics, which operate as follows:
- **Confidence Measure:** The confidence measure evaluates how well the GM can classify a given RSS fingerprint based on its class activations. For each sample (x), the GM generates class activations (shown in green in Figure 90), which represent the probabilities assigned to each class. The key metric used in this measure is the confidence score (CS), defined as the difference between the highest and second-highest class activations. Samples with low CS are prioritized for inclusion in the RM, as they indicate uncertainty in classification and are likely to be critical for improving the GM's performance. The CS is computed using Equation 58:

$$CS(x) = p_1(x) - p_2(x) \quad (58)$$

where $p_1(x)$ and $p_2(x)$ are the top two predicted probabilities for RSS fingerprint sample x .

- **Sensitivity Measure:** The sensitivity measure assesses the impact of each sample (x) on the GM's parameters (such as weights) by introducing small perturbations to the GM. This perturbation is calculated using the Frobenius norm of the weight difference between the baseline GM weights—before training on new RPs and the updated GM weights—after training on new RPs. The Frobenius norm determines the magnitude of the perturbation, ensuring that larger changes in the GM's weights result in proportionally larger perturbations, as described in Equation 59:

$$W_{perturbed} = W + \|W - W^b\|_F * N(0,1) \quad (59)$$

The perturbed GM is then evaluated to observe changes in its class activations. Samples that cause significant changes in the activations are identified as highly sensitive, indicating their importance for maintaining the GM's stability and generalization capabilities. In Equation 59 W represents the updated GM weights after training on new RP, W^b represents the baseline GM weights before training on new RP, $\|W - W^b\|_F$ is the Frobenius norm of the weight difference, and $N(0,1)$ is noise from a standard normal distribution, which introduces variability to simulate diverse perturbations, ensuring robustness and preventing deterministic bias in sensitivity analysis. The Frobenius norm $\|W - W^b\|_F$ is computed as:

$$\| W - W^b \|_F = \sqrt{\sum_i \sum_j (W^{ij} - W^{b,ij})^2} \quad (60)$$

The sensitivity of a sample (x) is evaluated using the distillation loss, measured as the kullback-leibler (kl) divergence between the gm’s predictions before and after applying the perturbation, as described in Equation 61:

$$SS(x) = KL(f_{GM}(x) \| f_{perturbed\ GM}(x)) \quad (61)$$

The KL divergence measures how much one probability distribution diverges from another. In our problem context, it reflects how sensitive the GM’s predictions are to changes in the weights caused by the perturbation. A higher KL divergence indicates that the sample significantly influences the GM’s predictions, marking it as highly sensitive. Samples with the highest sensitivity $SS(x)$ are prioritized for inclusion in the RM because they play a crucial role in preserving the GM’s stability across both old and new data. High-sensitivity samples prevent the GM from converging to local minima, which optimize for learning new data at the cost of forgetting patterns learned from old data. Instead, they act as stabilizers, anchoring the GM closer to the global minima, where performance is balanced across both old and RPs. This ensures robust learning, preventing catastrophic forgetting while integrating new knowledge effectively.

By leveraging confidence and sensitivity scores, CIELO ensures that the RM is populated with the most critical and informative samples. Selecting the k least confident samples ensures that uncertain samples, which can benefit most from retention, are prioritized, addressing data storage challenges by reducing the need to store all data. At the same time, selecting the k most sensitive

samples helps mitigate catastrophic forgetting by retaining key samples that stabilize the GM and maintain its ability to generalize across both old and new RPs. This dual k sample selection strategy optimizes memory usage while ensuring robust CIL for evolving indoor localization tasks.

- **Memory Optimization:** The memory optimization process addresses the challenge with the RM growing excessively large as more RPs are added. This process ensures that the memory remains compact and efficient by eliminating redundant or less informative samples while retaining diversity and representativeness in the data. CIELO defines a memory budget (M_b), which specifies the maximum size of the RM. This budget is critical to ensure efficient memory utilization, particularly in resource-limited environments. It prevents excessive memory usage while maintaining enough impactful samples to support the GM's generalization across both old and new RPs. When the memory buffer reaches M_b , CIELO prioritizes the removal of high-confidence, low-sensitivity samples per RP, as these are well-learned and contribute less to further learning. Retaining too many high-confidence samples can also lead to overfitting, where the GM becomes overly specialized to specific instances and struggles to generalize to new or unseen data. However, to avoid catastrophic forgetting, CIELO guarantees that at least one sample per RP is retained, even if it is a high-confidence sample. This safeguard ensures that no RP is completely removed, preserving the GM's ability to generalize across all RPs. By periodically pruning less critical samples while adhering to M_b , CIELO balances memory efficiency and generalization, supporting robust CIL without exceeding storage constraints.

10.6. EXPERIMENTAL SETUP

To evaluate the proposed CIELO framework, experiments were conducted in two buildings with distinct layouts: Building 1, containing 60 RPs and up to 193 visible APs, and Building 2, containing 90 RPs and up to 78 visible APs. Each RP was spaced at a 1-meter interval to ensure adequate granularity for precise indoor localization. RSS values were standardized to a range between 0 dBm (strongest signal) and -100 dBm (weakest signal), maintaining consistency across varying signal strengths.

Testing was performed using two smartphone devices carried by users: HTC U11 and Samsung Galaxy S7. Self-device testing involved training and testing using data collected from the same device, while cross-device testing assessed the GM generalizability by training on data from one device and testing on the data from another device. During the offline phase, the initial GM was trained using data from 30% of the RPs in each building. The remaining 70% of RPs were incrementally introduced during the CIL phase. At each RP, six RSS fingerprints were collected for training and one RSS fingerprint was reserved for testing. The performance of CIELO was assessed using the Euclidean distance (ED) method as defined in Equation 62:

$$ED = \sqrt{(X_{pred} - X_{true})^2 + (Y_{pred} - Y_{true})^2 + (Z_{pred} - Z_{true})^2} \quad (62)$$

ED is used to measure the distance between predicted location $(X_{pred}, Y_{pred}, Z_{pred})$ and ground-truth location $(X_{true}, Y_{true}, Z_{true})$, offering a direct and interpretable metric for evaluating localization accuracy, where smaller distance values indicate better performance. It serves as a reliable measure to assess and compare CIELO's performance across different scenarios.

The GM used in CIELO is a Multi-layer Perceptron (MLP) with ReLU activations in each layer. We explored CIL for indoor localization across two separate scenarios in two different buildings. For Building 1, the GM consists of two hidden layers: the first hidden layer has 144 neurons, and the second hidden layer has 67 neurons. For Building 2, the first hidden layer has 58 neurons, and the second hidden layer has 27 neurons. The number of output layer neurons varies as a function of the changing number of RPs to be predicted over time. The GM resulted in a maximum of 77,541 total parameters for Building 1 and a maximum of 14,801 parameters for Building 2. Offline training employed the Adam optimizer with a 0.01 learning rate using sparse categorical cross-entropy loss, while incremental retraining employed a learning rate of 0.1.

10.6.1. SENSITIVITY ANALYSIS: DETERMINING SAMPLE SELECTION SIZE (K)

In our CIL framework, selecting the number of samples per class (k) to preserve in the RM is crucial for balancing memory usage and localization performance. To determine the optimal k , we experimented with saving 0 to 6 samples per class in the RM and analyzed localization accuracy. As shown in Figure 91, increasing k reduces localization error by providing the GM with a broader representation of signal variations, enhancing generalization. However, improvements diminish beyond $k=4$. Retaining 5 or 6 samples slightly improves accuracy but requires higher memory overheads. Choosing $k=4$ strikes a balance between memory usage and performance, offering significant accuracy gains while keeping memory manageable. Beyond $k=4$, additional samples yield marginal accuracy improvements that do not justify the increased resource usage. For instance, Building 1 benefits slightly from $k=5$ or $k=6$, but the trade-off makes $k=4$ more practical.

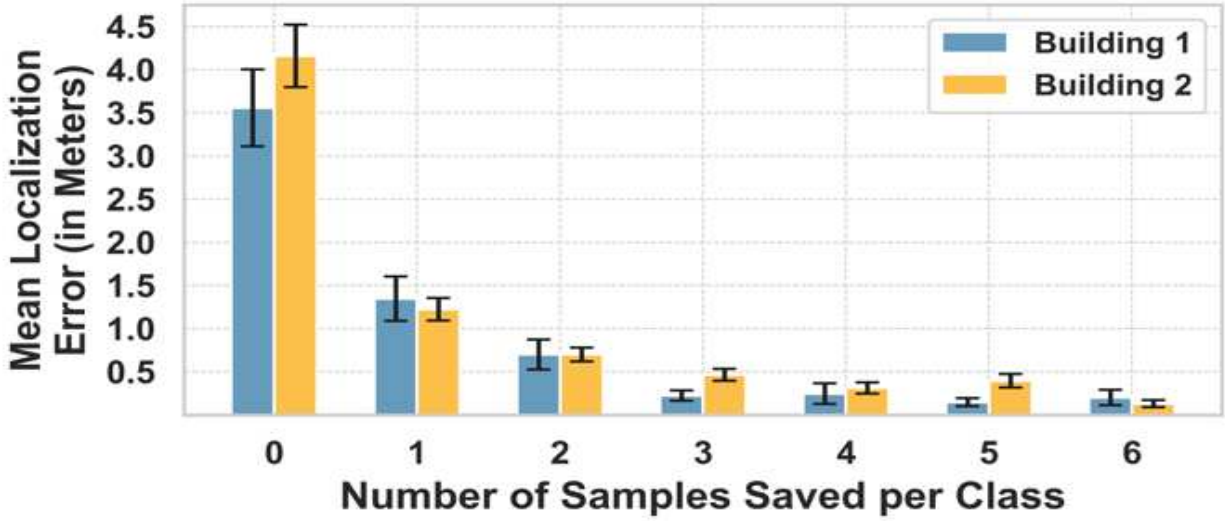


Figure 91: Mean localization error with increasing number of samples per class

10.6.2. SENSITIVITY ANALYSIS: DETERMINING MEMORY BUDGET (M_b)

The memory buffer size (M_b) directly impacts the trade-off between memory storage in RM and GM performance in CIL. To evaluate its impact, we conducted experiments with M_b allocations set to 0%, 25%, 50%, 75%, and 100% of the total memory required to store all samples for a given floor plan, analyzing their effect on mean localization error across the two buildings and testing devices.

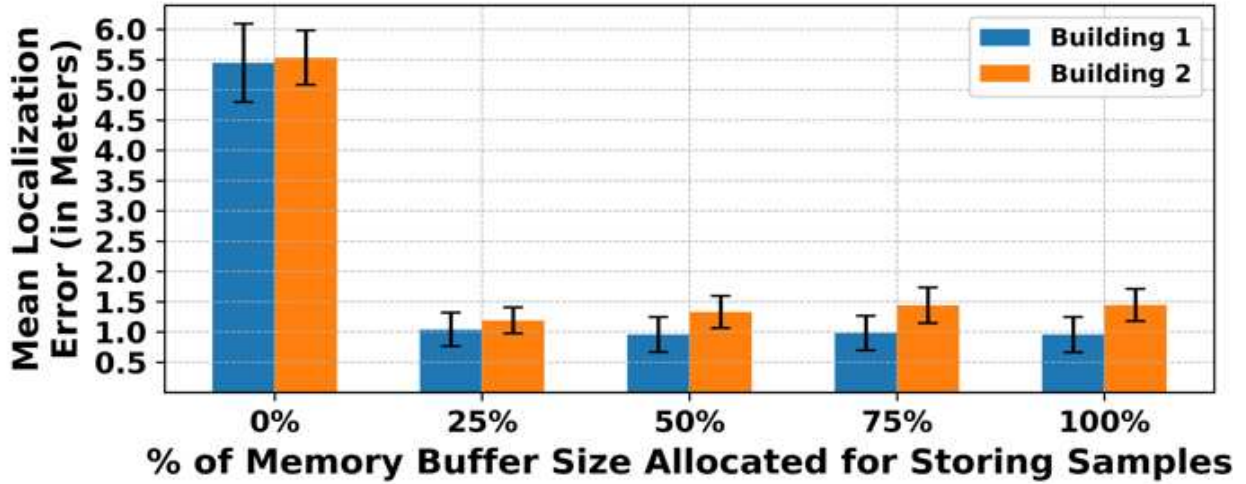


Figure 92: Mean localization error with increasing M_b

As shown in Figure 92, increasing M_b improves accuracy, with the highest error at 0% due to the absence of representative data for earlier RPs. In Building 1, mean error drops significantly at 25% allocation, with diminishing returns beyond this point. This reflects CIELO's memory optimization strategy, where retaining only the most impactful and diverse samples enhances performance. In Building 2, a more gradual improvement in error reduction suggests that the environment may require a larger memory allocation to adequately represent the signal diversity. However, increasing M_b beyond a certain point risks redundancy and inefficiency. At 100% allocation for instance, the error is slightly higher compared to 25%-75% due to overfitting caused by retaining redundant high-confidence, low-sensitivity samples that do not contribute meaningfully to generalization. These redundant samples can cause the GM to become overly specialized, reducing its ability to generalize to new or unseen data. This issue is more evident in Building 2, which undergoes more retraining cycles due to its larger number of RPs. Allocating 25% of memory, paired with 4 samples per class, strikes a balance by preventing overfitting,

optimizing RM use with 75% reduction in data storage, and maintaining high localization accuracy.

10.6.3. COMPARISONS WITH STATE-OF-THE-ART

To evaluate CIELO’s performance against state-of-the-art CIL and indoor localization frameworks, we conducted a comparative study involving Elastic Weight Consolidation (EWC) [196], Incremental Classifier and Representation Learning (iCaRL) [200], and Learning Without Forgetting (LwF) [198], which represent diverse state-of-the-art CIL strategies from prior work. We also compared against FEDHIL [88], which supported distributed and crowdsourced learning for indoor localization. The results of this study are shown in Figure 93 (for Building 1) and Figure 94 (for Building 2). The experiments were performed across the two buildings on paths with different layouts, using Samsung Galaxy S7 and HTC U11 devices to assess both self and cross-device testing. Mean localization error was measured as a function of the cumulative number of RPs learned by the GM to determine forgetting rate, with new RPs introduced in increments of 5.

Across all configurations in Building 1 and Building 2, for self-device and cross-device testing, CIELO consistently outperformed other frameworks, achieving lower mean localization errors as more RPs were learned. CIELO’s RMM policy allows dynamic sample selection based on confidence and sensitivity, enabling it to integrate new RPs effectively while retaining accuracy, demonstrating strong resilience to catastrophic forgetting.

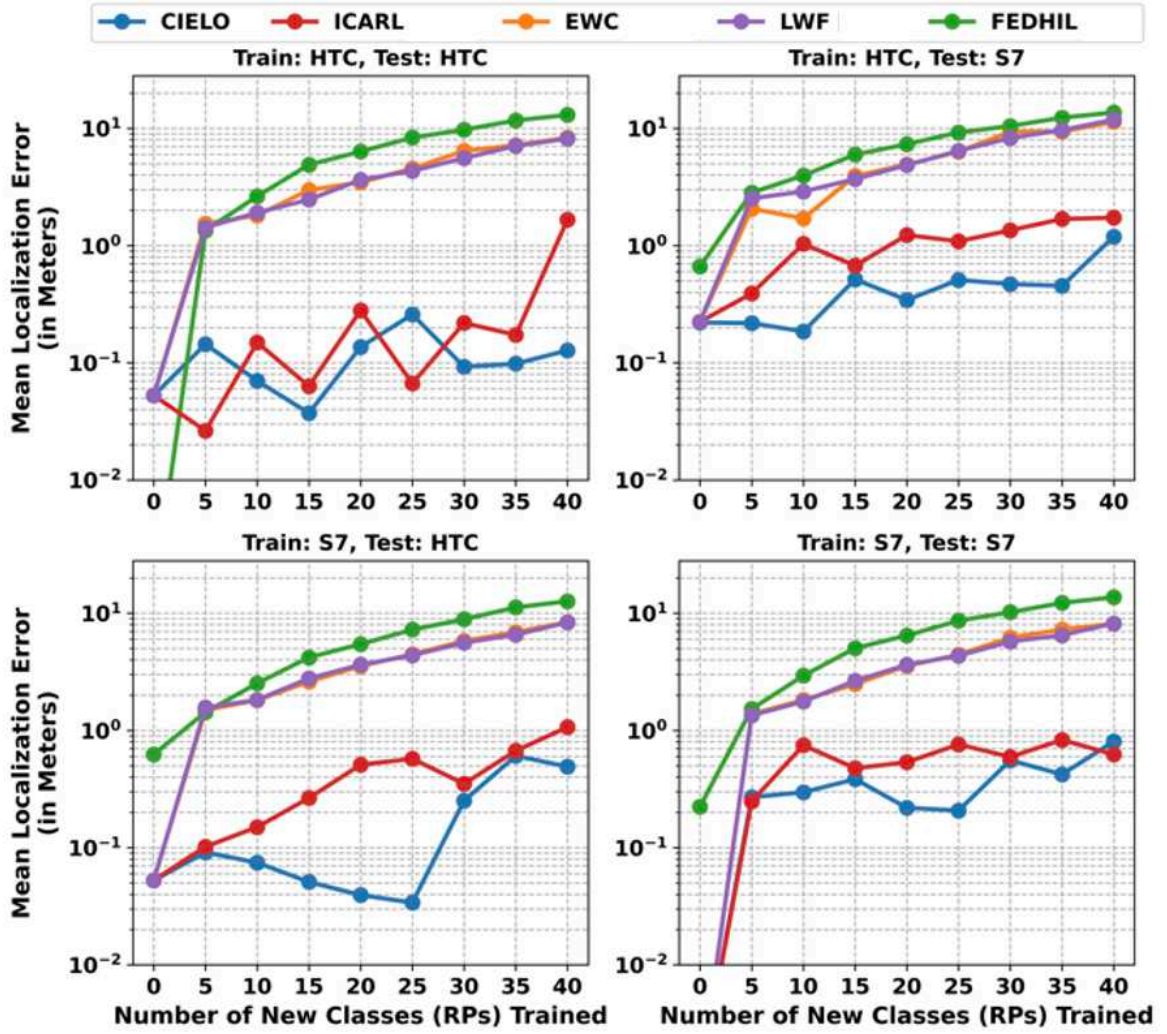


Figure 93: Comparison of mean localization errors across different state-of-the-art frameworks under both self and cross-device testing for Building 1.

In Building 1, as shown in Figure 93, with self-device testing (S7), CIELO maintained significantly lower localization errors compared to other methods, which showed a larger increase in error as new RPs were learned. In cross-device testing (training on S7, testing on HTC), CIELO exhibited strong generalization, maintaining consistent accuracy on a different device. In contrast, EWC [196], iCaRL [200], and LwF [198] struggled with higher errors, indicating challenges with non-IID data and incremental adaptation.

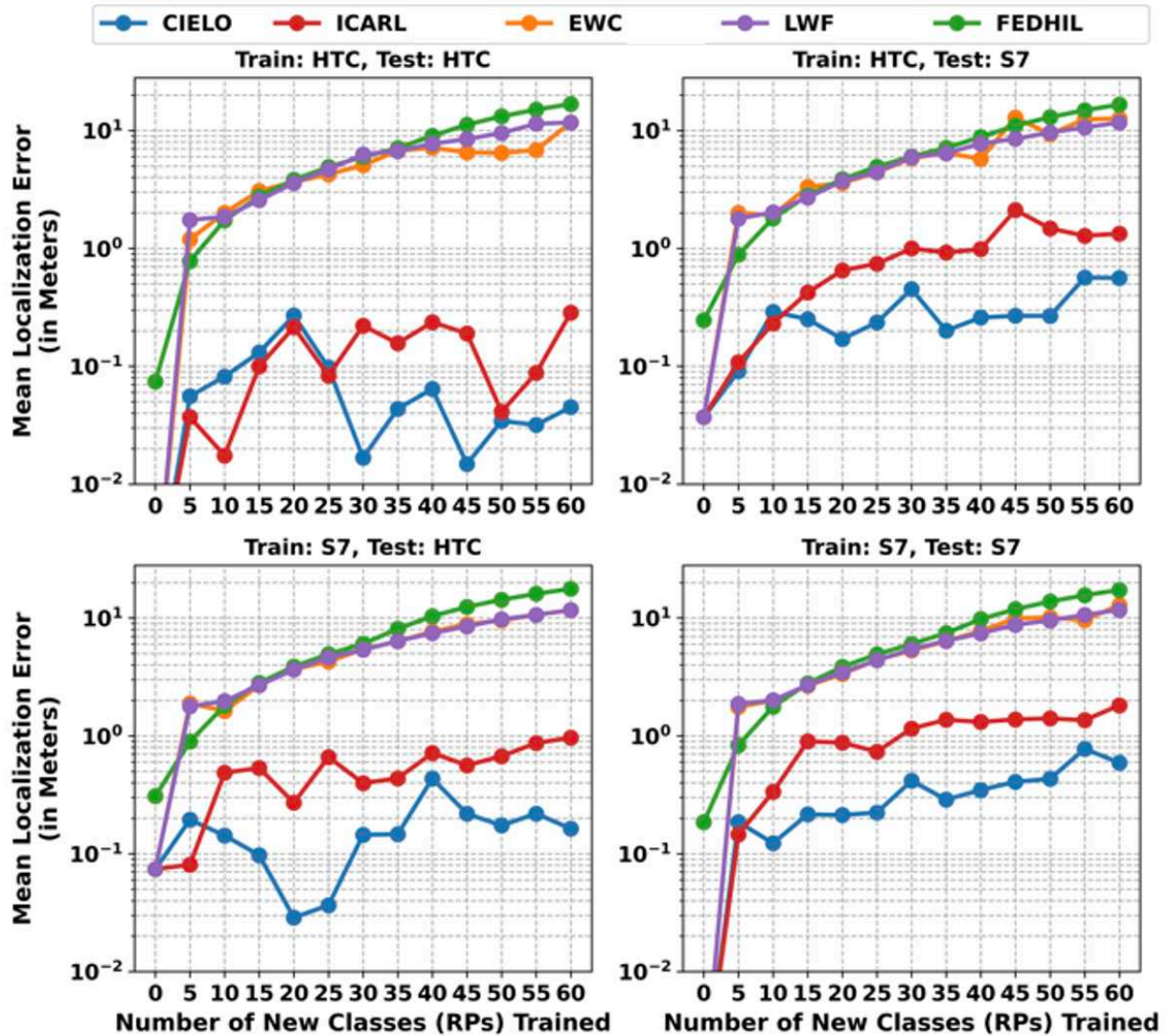


Figure 94: Comparison of mean localization error across different state-of-the-art frameworks under both self and cross-device testing for Building 2.

In Building 2, as shown in Figure 94, with its higher number of RPs, CIELO's advantage persisted, as it maintained a steady, low-error trajectory while other frameworks displayed progressively higher mean localization errors. These results highlight CIELO's robust RMM

policy, making it an effective solution for real-world indoor localization tasks with class-incremental learning requirements.

The results from Figure 93 and Figure 94 are more concisely summarized in Figure 95. CIELO achieved a mean localization error of 0.32 meters and a worst-case error of 0.4 meters, representing up to $29.4\times$ improvement over state-of-the-art methods. In comparison, iCaRL, the next best method, showed a mean error of 0.77 meters, while EWC [196], LwF [198], and FEDHIL [88] demonstrated mean errors up to 9.75 meters and worst-case errors as high as 12.54 meters. The poor performance of these baselines can be attributed to their inherent limitations. EWC’s rigid weight constraints restrict adaptation to new RPs, while iCaRL’s similarity-based sample retention fails to capture the diversity needed for generalization. LwF struggles with inconsistent task outputs in non-IID environments, as it uses TIL-based training, and FEDHIL lacks an efficient mechanism for incremental learning. CIELO’s ability to mitigate these challenges through strategic RM optimization and prioritization of impactful samples establishes it as a robust and efficient solution for CIL in dynamic indoor environments.

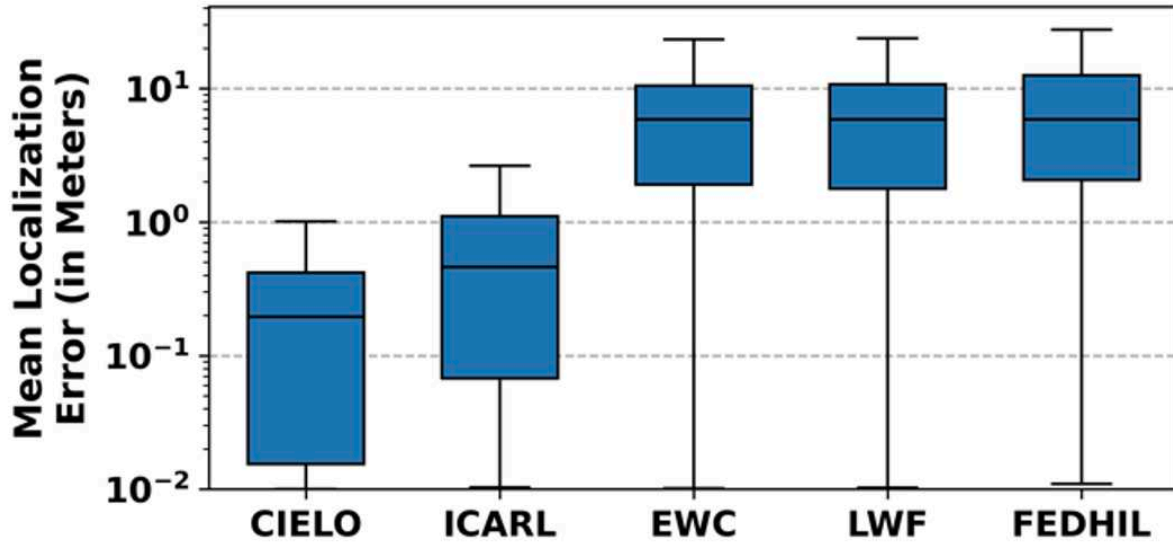


Figure 95: Comparison of CIELO against the state-of-the-art.

10.6.4. IMPACT OF NEW CLASS INCREMENT SIZE ON PERFORMANCE

To assess adaptability to varying class increments and retraining efficiency, we evaluated mean localization error and retraining loss for increments of 2, 5, and 10 new RPs, again comparing CIELO with the frameworks from iCaRL [200], LwF [198], EWC [196], and FEDHIL [88]. The results are shown in Figure 96. Across all methods, larger class increments (5 and 10 RPs) resulted in lower localization errors compared to smaller increments (2 RPs). This trend reflects the advantage of fewer retraining cycles with larger increments, which reduce disruptions to previously learned representations and mitigate catastrophic forgetting. CIELO consistently achieved the lowest errors, leveraging memory optimization and sensitivity-based sample selection to retain critical data and maintain generalization. In contrast, baseline methods struggled due to their inability to efficiently adapt to frequent updates and incremental learning in non-IID settings, resulting in higher localization errors.

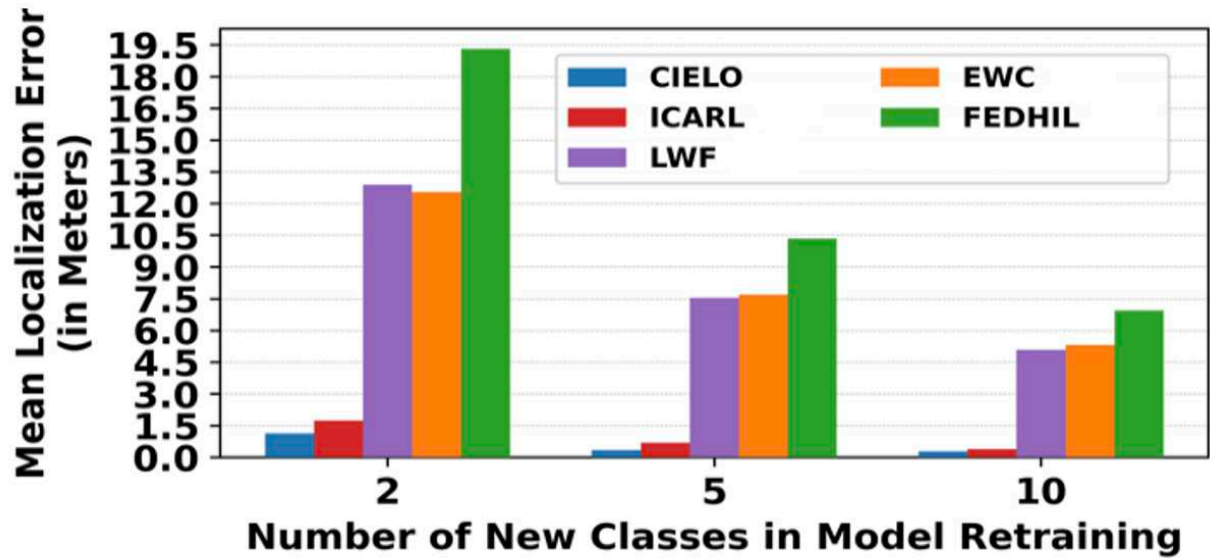


Figure 96: Mean localization error with varying number of new classes (RPs) introduced through retraining.

Figure 97 shows the average retraining loss across all buildings and testing devices for the three scenarios explored in Figure 96. CIELO demonstrated faster convergence and lower retraining losses by prioritizing impactful samples, effectively mitigating catastrophic forgetting and outperforming other frameworks.

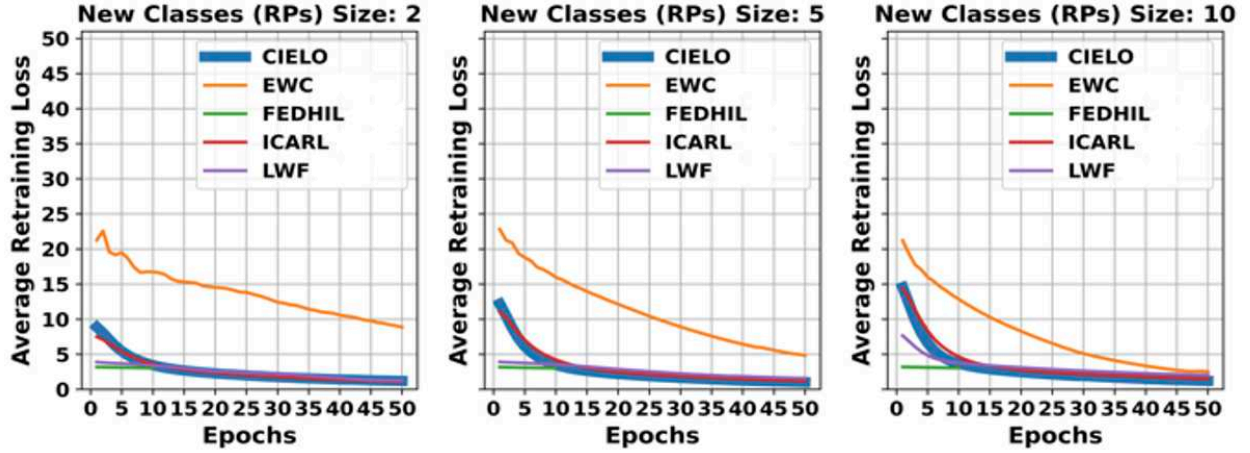


Figure 97: Average retraining loss with varying number of new classes (RPs) introduced through retraining.

10.6.5. EVALUATION OF COMPUTATIONAL COST ON EDGE SERVER

Lastly, the computational cost of each framework was evaluated in terms of % CPU usage and power consumption during continual retraining on an edge server that is equipped with an AMD Ryzen 7 5800HS processor. Table 18 summarizes the results for the retraining across the scenarios depicted in Figure 93 and Figure 94, averaged over the two buildings considered. Also shown for reference is the mean localization error for each framework.

It can be observed that CIELO achieves the best balance, with a low mean localization error of 0.32 meters and efficient resource usage (24.92% CPU, 18.72W). In contrast, EWC and FEDHIL exhibited high CPU usage (over 54%) and power consumption (~29W). EWC's fixed weight constraints require frequent computations to preserve prior knowledge, increasing overhead, while FEDHIL's GM aggregation technique involves extensive updates, making it resource intensive. iCaRL, with moderate resource usage, benefits from its selective replay mechanism but still incurs costs due to replay computations. LwF, while resource-efficient

(17.56W, 21.6% CPU), uses task-output preservation, requiring fewer computations but sacrificing accuracy due to disruptions in dynamic environments. These results highlight CIELO’s ability to reduce edge server resource utilization while maintaining high localization performance, making it suitable for resource-limited and evolving indoor scenarios.

Table 18: CPU usage, average power, and mean localization error comparison across frameworks

Framework	CPU Usage	Average Power	Mean Localization Error
CIELO	24.92 %	18.72 W	0.32 Meters
ICARL	23.38 %	18.18 W	0.77 Meters
LWF	21.6 %	17.56 W	7.02 Meters
EWC	54.5 %	29.08 W	6.99 Meters
FEDHIL	54.2 %	28.97 W	9.75 Meters

10.7. CONCLUSION

In this chapter, we introduced CIELO, a novel Class-Incremental Learning (CIL) framework tailored for Wi-Fi RSS-based indoor localization. CIELO is designed to address the challenges of learning new indoor areas efficiently while mitigating catastrophic forgetting and reducing data storage overhead. To the best of our knowledge, CIELO is the first framework designed to support continual learning and CIL for indoor localization. By leveraging crowdsourced data and implementing a novel representation memory management (RMM) policy, CIELO seamlessly integrates new indoor locations while retaining knowledge of previously learned locations. Experimental results highlight CIELO’s effectiveness, demonstrating up to $29.4\times$ improvement in localization accuracy (while learning new classes), up to $1.75\times$ reduction in data storage requirements, and up to $1.69\times$ lower device power consumption compared to state-of-the-art approaches.

CHAPTER 11

DAILOC: DOMAIN-INCREMENTAL LEARNING FOR INDOOR LOCALIZATION USING SMARTPHONES

Wi-Fi fingerprinting-based indoor localization faces significant challenges in real-world deployments due to domain shifts arising from device heterogeneity and temporal variations within indoor environments. Existing approaches often address these issues independently, resulting in poor generalization and susceptibility to catastrophic forgetting over time. In this chapter, we propose DAILOC, a novel domain-incremental learning framework that jointly addresses both temporal and device-induced domain shifts. DAILOC introduces a novel disentanglement strategy that separates domain shifts from location-relevant features using a multi-level variational autoencoder. Additionally, we introduce a novel memory-guided class latent alignment mechanism to address the effects of catastrophic forgetting over time. Experiments across multiple smartphones, buildings, and time instances demonstrate that DAILOC significantly outperforms state-of-the-art methods, achieving up to $2.74\times$ lower average error and $4.6\times$ lower worst-case error.

11.1. INTRODUCTION

The indoor localization market is projected to reach USD 84.17 billion by 2030, as demand rises for precise positioning [213]. While GPS/GNSS has transformed outdoor positioning, indoor localization remains a difficult problem due to the absence of satellite signals and the complexity of indoor environments. Indoor Localization aims to estimate the location of a device or user within a building, providing crucial functionality in homes, offices, shopping centers, hospitals, and industrial settings. A widely used approach for indoor localization is Wi-Fi-based Received Signal

Strength (RSS) fingerprinting [161], [214]. In this approach, during an offline phase, Wi-Fi RSS measurements are collected at fixed locations (aka Reference Points or RPs) to build a fingerprint database. Often, a machine learning (ML) model is trained to map these fingerprints to known locations. In the online phase, the model uses real-time RSS measurements to predict the user's position based on the previously learned mappings.

However, in real-world deployments indoor environments are rarely static. Factors such as moving furniture, changing layouts, or shifting occupancy patterns can significantly alter signal characteristics [135]. This mismatch between training and deployment conditions, known as domain shift, arises from two major factors in indoor localization: temporal variations (e.g., furniture rearrangement, occupancy changes, seasonal effects) and device heterogeneity (e.g., different smartphones with varying hardware and firmware). These shifts disrupt signal patterns, causing ML models trained offline to degrade significantly over time or across devices.

The impact of such domain shifts is illustrated in Figure 98. Here, the state-of-the-art CNRP model [164] is trained on data collected from 60 RPs using a BLU Vivo 8 (BLU) device at time = 0 and evaluated on the BLU, Samsung Galaxy S7 (S7), and HTC U11 (HTC) devices over time. The left plot shows that even on the training device (BLU), localization error increases sharply over time, demonstrating the effect of temporal domain shift. The right plot shows the model's performance on the three different devices at the same time point as the training instance, highlighting a domain shift caused by device heterogeneity. These results show how domain shifts, whether due to time or hardware, can significantly degrade model reliability in real-world settings.

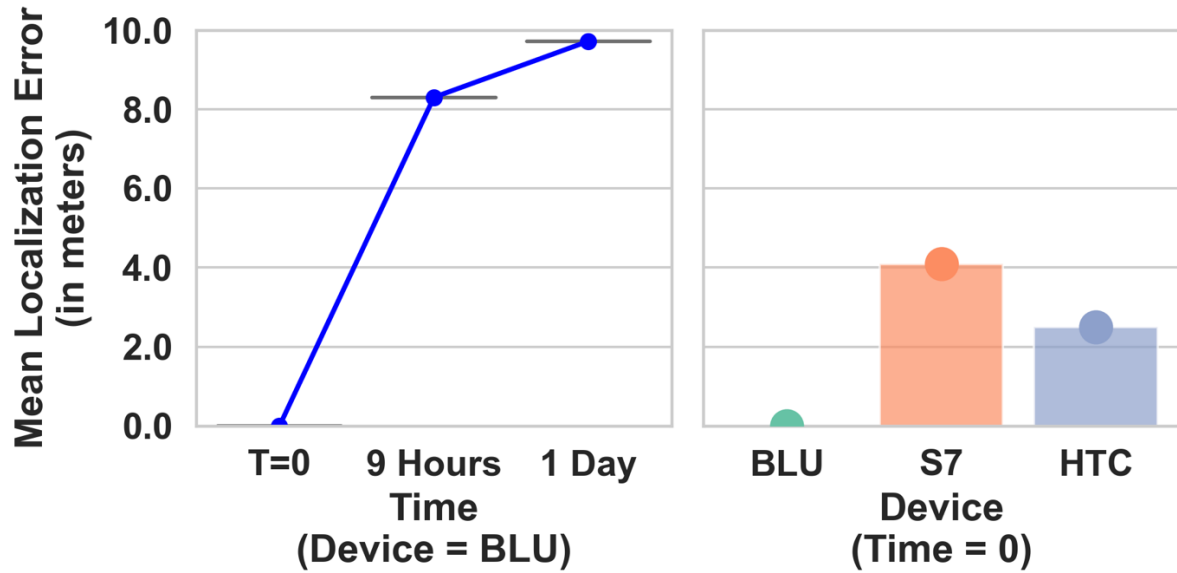


Figure 98: Illustration on impact of domain shifts arising from device heterogeneity and temporal variations on indoor localization performance.

Current solutions typically rely on either retraining the model whenever accuracy drops, which requires costly re-collection of labeled data, or using adaptive learning strategies such as Federated Learning (FL) [215], where the model's own predictions are treated as labels for incoming unlabeled data. While these adaptive learning methods reduce the need for manual labeling, they introduce two major challenges: pseudo-label noise [216], where incorrect predictions are treated as ground truth during adaptation, and catastrophic forgetting [197], where learning on new data causes the model to lose previously acquired knowledge about earlier environments or known devices. These challenges make it difficult to sustain reliable indoor localization accuracy under persistent temporal and device-related domain shifts.

11.2. CONTRIBUTIONS

To address these important challenges posed by temporal and device domain shifts, we propose DAILOC, a novel domain-incremental learning framework for efficient Wi-Fi-fingerprinting based indoor localization in dynamic indoor environments. The key contributions of this work are:

- A domain-incremental machine learning framework (DAILOC) that addresses dynamic device and temporal (environment) shifts by enabling continual adaptation.
- A disentangled representation learning strategy which improves generalization while reducing pseudo-label noise during unsupervised adaptation.
- A memory-guided class latent alignment mechanism that mitigates catastrophic forgetting by statistically aligning latent embeddings across sequential domains.
- Extensive evaluations on real-world RSS data, demonstrating improved accuracy and stability compared to state-of-the-art frameworks.

11.3. RELATED WORK

Wi-Fi fingerprinting-based indoor localization has gained significant prominence in recent competitions hosted by IPIN [199] and Microsoft [34]. Early ML-based approaches, including KNN [201], HMM [48], and GPC [202], aimed to address fluctuations in RSS caused by human movement, and multipath effects but struggled to generalize across diverse environments due to their shallow architectures [203]. More recent deep learning (DL) models such as DNNLOC [121], CNNLOC [28], and ANVIL [45] have improved localization accuracy across diverse environments through learned feature extraction. Among these, ANVIL stands out by providing calibration-free and device-invariant localization using attention-based neural networks. However,

ANVIL and other models are trained once and deployed statically, lacking the ability to adapt to unknown devices or environmental changes over time. This leaves them vulnerable to performance degradation in dynamic, real-world settings.

To address concerns with real-world deployment, federated learning (FL) approaches have been explored as a means to enable models to adapt continuously while preserving user privacy. For example, FedLoc [175] applies the popular FedAvg algorithm to collaboratively train a shared global model across heterogeneous clients, while FedHIL [88] introduces impact-driven aggregation to better handle client heterogeneity. CNRP [164] reduces reliance on labeled data through pseudo-labeling but suffer from noisy labels and unstable updates under domain shifts. In contrast, our approach in this work leverages pseudo-labels with strategies to suppress noise and enable long-term robust continual adaptation across evolving devices and indoor environments.

Beyond pseudo-labeling, a few domain adaptation techniques have been explored to handle domain shifts by aligning feature spaces between a source and target domain. Early approaches, such as signal strength difference and hyperbolic location fingerprinting [217], focused on calibration-free signal-level adjustments to reduce device heterogeneity. More recent deep learning-based methods aim to learn domain-invariant representations directly from data. For example, FIDORA [162] employs VAE-based augmentation and joint reconstruction-classification to improve adaptation with minimal labeling, while DAFI [218] aligns both marginal and conditional distributions for device-free localization. STELLAR [134] uses Siamese attention networks to handle temporal variations. TransLoc [219] introduces a cross-domain mapping framework that refines the source domain and constructs a shared latent space for heterogeneous feature transfer. While effective for isolated source–target settings, these methods are primarily

designed for static, pairwise adaptation and do not scale naturally to real-world deployments where new devices, environments, and users emerge continuously.

While domain-incremental learning has been explored in vision and time-series applications [220], [221], it remains underexplored in indoor localization. Recent work such as DARE [222] highlights the importance of mitigating representation drift during domain-incremental learning to prevent catastrophic forgetting. While DARE focuses on generic task sequences, it assumes access to clean task boundaries and labeled data, conditions that rarely hold in real-world indoor localization. This limitation highlights the need for a framework like DAILOC that supports incremental domain adaptation, sustaining localization accuracy over time despite device heterogeneity and temporal drift.

11.4. BACKGROUND: DOMAIN INCREMENTAL LEARNING FOR INDOOR LOCALIZATION

In real-world indoor localization systems, the relationship between Wi-Fi RSS and physical locations is sensitive to environmental changes. Over time, factors such as furniture movement, new access points (APs), or device variations alter the RSS patterns. This mismatch between training conditions and deployment conditions is known as domain shift. Unlike random noise, domain shifts represent deeper changes in signal distribution that can cause models to fail after deployment. Traditional domain adaptation methods try to adjust models trained in one setting (such as on a specific device or at a specific time) to work in another setting (like a different device or a later time). However, these approaches usually assume a one-time adaptation between two fixed domains. In practice, domain shifts are continuous and unpredictable within indoor

localization, as unknown devices may appear at any time, and indoor environments may also evolve gradually or suddenly, in different ways.

To address these challenges, our DAILOC framework implements a domain-incremental learning strategy, enabling continual adaptation without retraining from scratch. During deployment, DAILOC follows two adaptation paths depending on the device’s status. For an unknown device, labeled data is used for supervised onboarding. For a known device, unlabeled data from users is used for unsupervised adaptation, relying on pseudo-labels generated by the model itself. This setup reflects real-world deployment constraints, where collecting labeled samples for an unknown device once (e.g., by the localization solution provider) for onboarding is feasible. However, continuously collecting and manually labeling data across all time periods, environments, and devices quickly becomes impractical. Additionally, storing large amounts of data from all past domains would violate privacy guarantees and impose heavy memory and management overheads, especially in resource-limited mobile systems. Thus, the model must adapt incrementally using only current data, without retaining full access to prior domain samples. This realistic setup introduces two key challenges:

- **Catastrophic forgetting:** where adapting to new unknown devices degrades prior knowledge.
- **Pseudo-label noise:** where domain shifts cause incorrect self-predictions on known devices, harming model updates during unsupervised adaptation.

To address these challenges, it is essential to disentangle domain-specific variations from location-relevant features. Disentanglement isolates factors such as device biases and temporal

environmental changes, allowing the model to focus adaptation on domain-specific components while preserving invariant features critical for reliable localization across evolving conditions. In the next section, we discuss DAILOC, with details of how it implements these principles using multi-level variational autoencoder (ML-VAE), disentangled representation, and memory-guided class latent alignment.

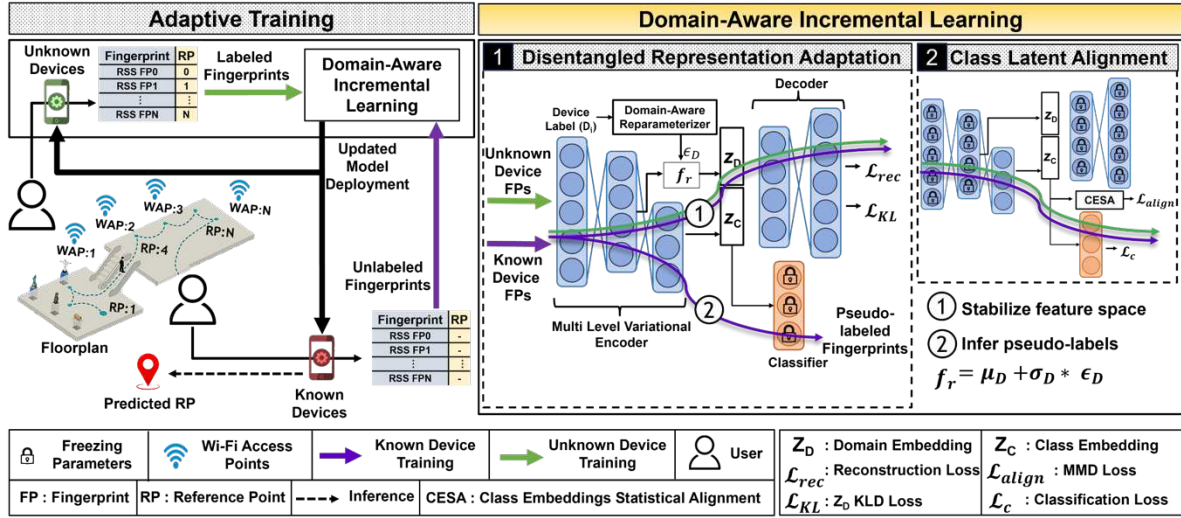


Figure 99: Overview of the DAILOC framework using domain-aware incremental learning for robust Wi-Fi RSS fingerprinting-based indoor localization

11.5. DAILOC FRAMEWORK

To meet the challenges of operating under dynamic conditions, where unknown devices appear and indoor environments evolve over time, we propose DAILOC, a domain-incremental learning framework (Figure 99) that enables models to adapt continually without requiring access to earlier domain data. DAILOC performs adaptive training which consists of two main stages: offline pretraining and online domain-aware incremental learning.

In the offline phase, DAILOC is trained on a curated labeled dataset, where RSS fingerprints are collected from various RPs using a fixed device under controlled conditions. These fingerprints

are mapped to known physical coordinates (RPs) and used to learn location-discriminative features. At the core of DAILOC is a multi-level variational autoencoder (ML-VAE) comprising an encoder and decoder, and paired with a lightweight classifier. The encoder extracts latent representations from RSS, the decoder reconstructs the inputs to retain signal characteristics, and the classifier maps location-relevant features to RPs. The model at this stage is jointly optimized to capture general RSS-location relationships under relatively stable device and environmental condition providing a strong initialization before encountering the domain variability present during deployment. Once deployed in the online phase, DAILOC continuously collects crowdsourced RSS fingerprints from end-user devices and performs real-time localization. Unlike the curated offline dataset, these online samples often lack ground-truth location labels and may originate from unknown devices or environments undergoing gradual changes.

To adapt to dynamic domain data, DAILOC includes a domain-aware hybrid incremental learning pipeline composed of two key stages: 1) Disentangled Representation Learning and 2) Class Latent Alignment. In the first stage, incoming RSS fingerprints are encoded through the ML-VAE that separates the features into location-relevant and domain-specific factors. A domain-aware reparameterization mechanism ensures consistent modeling of domain variations while preserving localization-relevant information. Once the domain feature space is stabilized, pseudo-labels are generated for unlabeled data from users using a frozen classifier. By adapting the encoder before label generation, DAILOC reduces pseudo-label noise caused by domain shifts. In the second stage, a Class Embedding Statistical Alignment (CESA) module aligns the pseudo-labeled latent representations from the current domain with prototypes stored from previous domains. This alignment is guided by a Maximum Mean Discrepancy (MMD) loss to maintain consistency without accessing previous raw data.

Through this two-stage disentanglement and alignment based incremental learning pipeline, DAILOC achieves robust domain-aware incremental learning across temporal and device shifts without explicit retraining. Each stage of this incremental learning pipeline is described next.

11.5.1. DISENTANGLED REPRESENTATION ADAPTATION

Effective domain-incremental adaptation requires separating device and environment-specific variations from stable location-relevant features. To achieve this, *DAILOC* employs a ML-VAE that disentangles input RSS fingerprints from both known and unknown devices into two latent spaces: z_C for capturing domain-invariant, location-relevant class features and z_D for capturing domain-specific variations arising from device or environmental shifts (Figure 100). This disentangled representation allows adaptation to new domains by adjusting for signal variations while preserving location information critical for accurate localization.

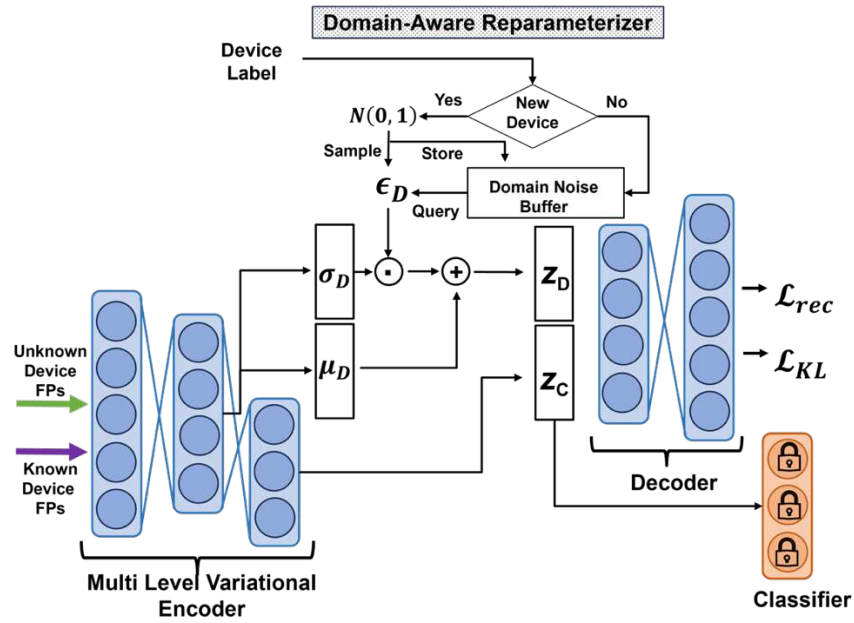


Figure 100: DAILOC's multi-level VAE with disentanglement representation

To achieve disentanglement as shown in Figure 100, a domain-aware reparameterization is used inside the ML-VAE. Inspired by grouped observation frameworks such as [223], the encoder predicts mean vector μ_D and standard deviation σ_D for the domain-specific latent space, but instead of applying random noise independently for each sample, a fixed noise vector ϵ_D for each domain D is stored in a Domain Noise Buffer. During reparameterization, the corresponding ϵ_D is queried and applied to compute the domain-specific latent representation, as shown in Equation 63:

$$z_D = \mu_D + \sigma_D * \epsilon_D \quad (63)$$

where $\epsilon_D \sim N(0,1)$. This consistent sampling forces z_D to capture coherent domain-level variations such as device biases or temporal drifts. To regularize the latent spaces, we apply two loss components. First, a Kullback–Leibler (KL) divergence loss is used to encourage the learned posterior $N(\mu, \sigma^2)$ to remain close to a standard normal prior $N(0, I)$ promoting smoothness and compactness, as defined in Equation 64:

$$\mathcal{L}_{KL} = D_{KL}(N(\mu, \sigma^2), N(0, I)) \quad (64)$$

Second, a reconstruction loss $\mathcal{L}_{rec}$ is used to minimize the difference between the input RSS fingerprint x and its reconstruction $\hat{x}$ from the concatenated latent representations $[z_D, z_C]$ as defined in Equation 65:

$$\mathcal{L}_{rec} = ||x - \hat{x}||^2 \quad (65)$$

This two-part regularization leads to tight intra-domain clusters in z_D , while z_C remains free from such structured noise and captures stable location-relevant features.

During adaptation, the classifier is frozen (see Figure 100), and only the encoder is updated. This ensures that z_D gradually adapts to evolving domains while preserving the existing decision boundaries in the classifier. Once the feature space stabilizes, pseudo-labels are inferred for unlabeled data from known devices using the frozen classifier to serve as inputs to the Class Latent Alignment stage (discussed next). By isolating domain-specific noise in z_D and freezing the classifier, *DAILOC* aligns its feature space before any label-driven updates occur, significantly reducing the risk of pseudo-label noise. This controlled adaptation enables the model to refine its representations without corrupting learned location mappings, resulting in improved generalization to device and temporal domain shifts while supporting seamless unsupervised adaptation.

11.5.1. CLASS LATENT ALIGNMENT

To prevent catastrophic forgetting and preserve location identity across evolving domains, *DAILOC* introduces a Class Embedding Statistical Alignment (CESA) mechanism, as illustrated in Figure 101. CESA aligns class embeddings from the current domain with historical prototypes which are location-relevant latent vectors z_r stored during supervised onboarding of new devices in a Representation Memory Buffer. When an unknown device is introduced, supervised onboarding is performed to generate labeled location-relevant representations z_C . Instead of storing all samples, reservoir sampling is used, which is a random selection process that guarantees maintaining at least one representative embedding z_C per reference point (RP). For known devices, similarity between the current pseudo-labeled embeddings z_C and historical prototypes z_r is enforced using a MMD loss as defined in Equation 66:

$$\mathcal{L}_{align} = || \frac{1}{n} \sum_{i=1}^n \phi(z_{c,i}) - \frac{1}{m} \sum_{i=1}^m \phi(z_{r,j}) ||_H^2 \quad (66)$$

where n and m represent the number of embeddings from the current domain and the reference memory buffer, respectively, and $\phi(\cdot)$ is the mapping to a reproducing kernel Hilbert space H , enabling the comparison of distributions in a high-dimensional feature space. This loss minimizes the distributional discrepancy between current and historical embeddings preventing catastrophic forgetting. During this alignment phase, the encoder and decoder remains frozen, ensuring that latent representations do not drift. Only the classifier is updated using pseudo-labeled samples with a supervised classification loss ($\mathcal{L}_c$).

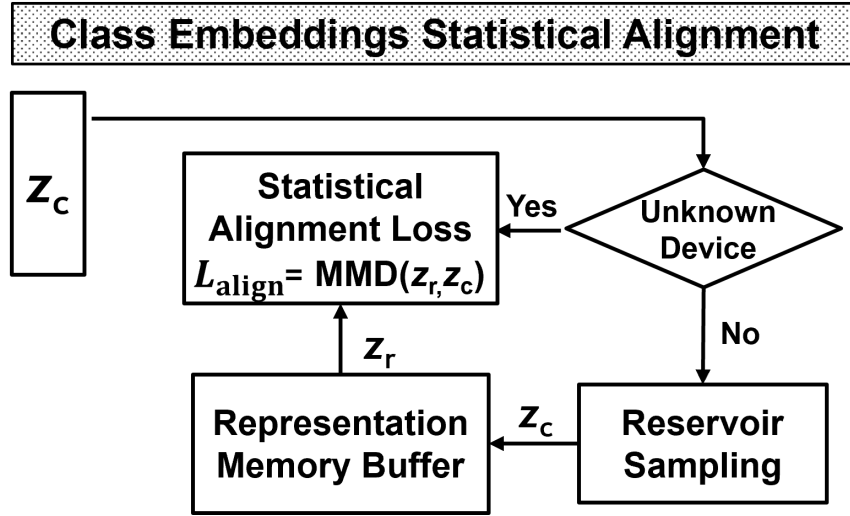


Figure 101: Statistical alignment of class embeddings using MMD loss to prevent catastrophic forgetting, as part of the CESA module.

11.5.3. JOINT DOMAIN-AWARE OPTIMIZATION

The synergy between memory-based alignment and disentangled representation ensures that *DAILOC* maintains the plasticity needed to adapt to unknown devices and the stability required to retain prior knowledge of know devices, while also supporting continuous adaptation to temporal

shifts. During adaptation, the outputs from both stages are optimized together. In the disentanglement stage, the reconstruction loss $\mathcal{L}_{rec}$ and KL divergence loss $\mathcal{L}_{KL}$ are minimized to stabilize and regularize the latent representations. After pseudo-labels are generated, the class alignment stage minimizes the MMD-based alignment loss $\mathcal{L}_{align}$ and the classification loss $\mathcal{L}_c$ to align embeddings and refine decision boundaries. These losses are summed up to form the total objective loss:

$$\mathcal{L}_{total} = \mathcal{L}_{rec} + \mathcal{L}_{KL} + \mathcal{L}_{align} + \mathcal{L}_c \quad (67)$$

This joint optimization enables DAILOC to mitigate catastrophic forgetting and achieve seamless, long-term integration of devices and environments under dynamic conditions. After adaptation, the updated encoder and classifier are deployed back for inference, ensuring improved localization performance in future predictions.

11.6. EXPERIMENTAL SETUP

We evaluate *DAILOC* in a realistic continual learning scenario under diverse scenarios that reflects both device heterogeneity and temporal signal drifts. Experiments were conducted in two buildings with distinct physical layouts and material compositions. Building 1 consists of 60 RPs and up to 193 visible Wi-Fi APs, featuring wood-cement construction, spacious open areas, and office infrastructure. Building 2 consists of 48 RPs and up to 168 Wi-Fi APs, with a modern metal–wood layout with dense shelving and heavy equipment, leading to strong signal reflection and attenuation effects. RPs were placed at 1-meter intervals to ensure fine-grained localization. RSS values were standardized to the range $[0, -100 \text{ dBm}]$ for consistency across devices and locations.

The *DAILOC* architecture for experiments is based on ML-VAE model paired with a lightweight classifier, jointly optimized to disentangle location-relevant and domain-specific features. The encoder consists of two hidden layers with 256 and 128 neurons (with ReLU activations), followed by two separate branches: one that generates the mean and variance for the 16-dimensional latent representation z_D , and the other that maps to the 16-dimensional domain-invariant latent representation z_C via a 64-unit intermediate layer. The decoder reconstructs input RSS vectors from the concatenated latent representations $[z_D, z_C]$ using two dense layers with 128 and 256 neurons, followed by a sigmoid output layer. The classifier operates on z_C and includes one 64-unit hidden layer and a SoftMax output layer for RP classification. The encoder, decoder, and classifier contain approximately 88K, 80K, and 89K parameters respectively. Training uses the Adam optimizer with a learning rate of 1×10^{-3} . Representation memory buffer size M is set to be equal to the number of RPs to accommodate at least one class prototype per RP.

To simulate device-level domain shifts, data was collected using six smartphones from different manufacturers (see Table 19). Devices varied in both hardware (Wi-Fi chipsets) and software (firmware-level signal processing). Even when chipsets were identical, manufacturer-specific differences in filtering introduced significant signal variation, creating diverse hardware and software heterogeneity conditions. To capture temporal dynamics, data was collected over six time instances, from initial offline training ($T=0$) to three months post-deployment reflecting environmental evolution, including occupancy changes and signal interference.

Table 19: Details of smartphones used in experiments

Manufacturer	Model	Acronym	Wi-Fi Chipset
BLU	Vivo 8	BLU	MediaTek Helio P10
HTC	U11	HTC	Qualcomm Snapdragon 835
Samsung	Galaxy S7	S7	Qualcomm Snapdragon 835
LG	V20	LG	Qualcomm Snapdragon 820
Motorola	Z2	MOTO	Qualcomm Snapdragon 820
OnePlus	3	OP3	Qualcomm Snapdragon 820

The performance of DAILOC was assessed using the Euclidean distance (ED) as defined in Equation 68:

$$ED = \sqrt{(X_{pred} - X_{true})^2 + (Y_{pred} - Y_{true})^2 + (Z_{pred} - Z_{true})^2} \quad (68)$$

ED was used to measure the distance between predicted location $(X_{pred}, Y_{pred}, Z_{pred})$ and ground-truth location $(X_{true}, Y_{true}, Z_{true})$, offering a direct and interpretable metric for evaluating localization accuracy, where smaller distance values indicate better performance.

11.6.1. COMPARISON WITH STATE-OF-ART

Figure 102 compares the mean localization error of DAILOC against three state-of-the-art indoor localization frameworks: DARE [222], FIDORA [162], and FEDHIL [88] (all discussed earlier in Section 9.3). In this experiment, one new device is introduced sequentially across different time instances for supervised onboarding. Once onboarded, each device undergoes unsupervised adaptation, with the study spanning a duration of three months. This setup enables a

comprehensive evaluation of DAILOC’s capabilities across three key aspects of plasticity, stability and temporal robustness.

The results in Figure 102 are averaged across Building 1 and Building 2 across the six time instances, from deployment time ($T=0$) to three months post-deployment. While all frameworks perform comparably at $T=0$, their performance diverges over time as domain shifts emerge. DARE [222], a domain-incremental learning-based method, demonstrates strong plasticity and effectively integrates new domains. However, in the context of indoor localization where ground truth is unavailable post-deployment, it relies on pseudo-labels for unsupervised learning. This leads to increased susceptibility to pseudo-label noise, reducing its stability over time. FIDORA [162], by contrast, is a domain adaptation approach focused on aligning features without retraining. It maintains better stability over time but lacks plasticity, as it cannot incorporate new devices or domain variations incrementally. FEDHIL [88], a federated learning-based method, provides moderate early performance but is prone to catastrophic forgetting in the long term. As a result, its accuracy degrades steadily over time. In contrast, DAILOC combines domain-incremental learning with domain-aware disentanglement and memory-guided class latent alignment. It preserves plasticity for integrating new devices and maintains stability during unsupervised adaptation, resulting in consistently lower error and reduced variance over time.

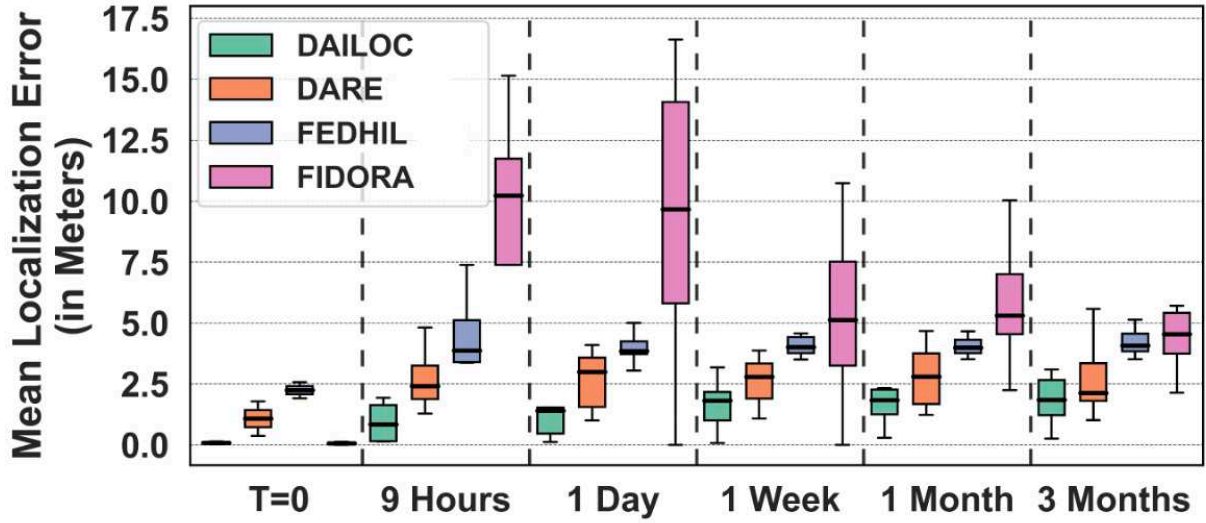


Figure 102: Comparison of DAILOC against state-of-the-art.

11.6.2. DAILOC PERFORMANCE ANALYSIS

In this next analysis, we present a more detailed set of results for our framework. Figure 103 and Figure 104 present mean localization error (in meters) across devices and time instances for Building 1 and Building 2 respectively, for DAILOC. Each heatmap reflects how localization error evolves as devices are incrementally introduced (as in the previous experiment) and the system adapts over time in an unsupervised manner.

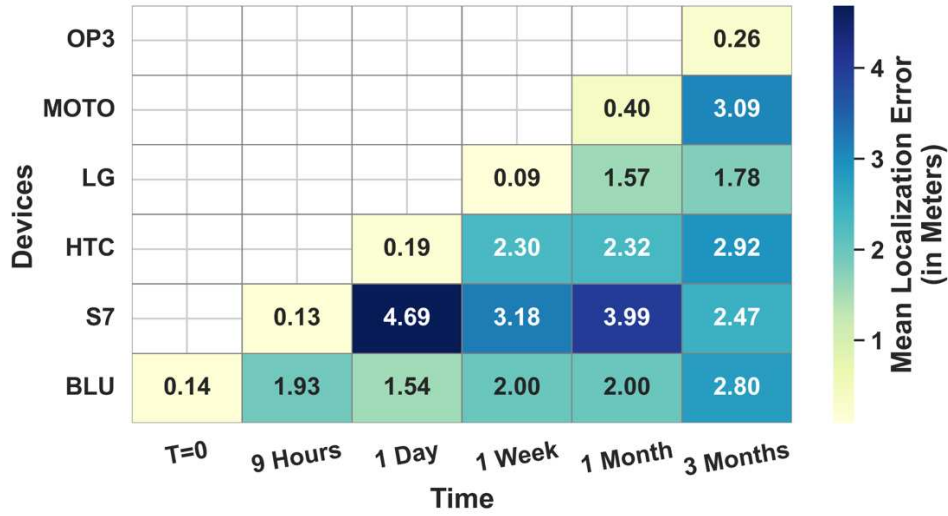


Figure 103: Comparison of mean localization error (in meters) for Building 1

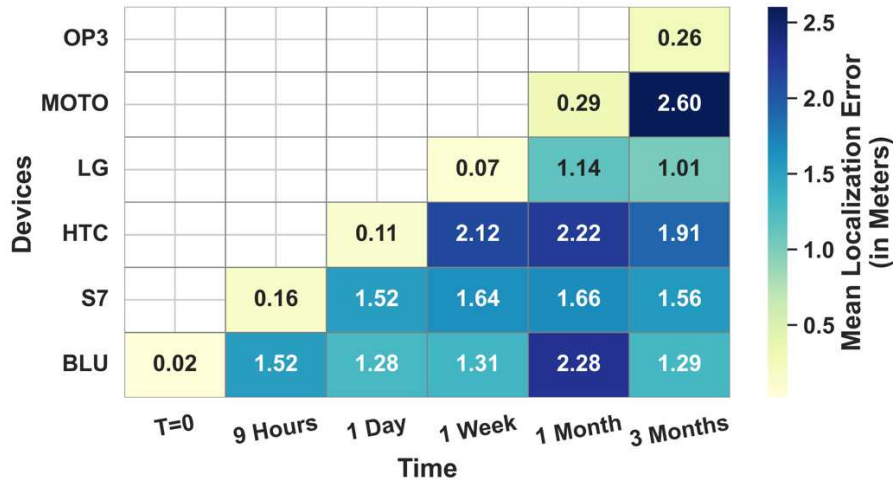


Figure 104: Comparison of mean localization error (in meters) for Building 2

DAILOC demonstrates strong localization performance in both buildings, despite their architectural and signal distribution differences. For instance, error remains under 2m for most devices even after 3 months of domain drift. In Building 2, which features more complex signal reflections due to metal shelving and open structures, DAILOC maintains performance more

effectively than in Building 1. For example, LG maintains a low error from week 1 to month 3 (1.14 m – 1.01 m), indicating resilience to longer-term drift. While devices like S7 and BLU show higher errors in Building 1 after longer time periods (e.g., S7: 3.99m error at month 1), DAILOC's overall performance remains significantly more stable compared to state-of-the-art methods (see Figure 103). These results across both buildings highlight that DAILOC's architectural design, rooted in disentangled representation and domain-incremental learning, is effective not only in adapting to new devices, but also in sustaining performance across varying spatial contexts and over time.

11.6.3. ABLATION STUDY: IMPACT OF DISENTANGLEMENT

To evaluate the role of disentangled representation learning on pseudo-label noise, we measure the mean pseudo-label error before and after disentanglement across five-time instances in both buildings. As shown in Figure 105, applying disentanglement significantly lowers pseudo-label error, especially in early time steps where domain shifts are more severe, and representations are poorly adapted. In Building 1, the error drops by over 80% at 9 hours (from 11.0 m to 1.5 m), and remains consistently lower throughout. A similar trend is observed in Building 2, with pseudo-label noise reduced by up to 60% across time instances. These results confirm that disentangling domain-specific features from location-relevant ones improves the reliability during unsupervised adaptation, making the framework more robust to noisy pseudo labels.

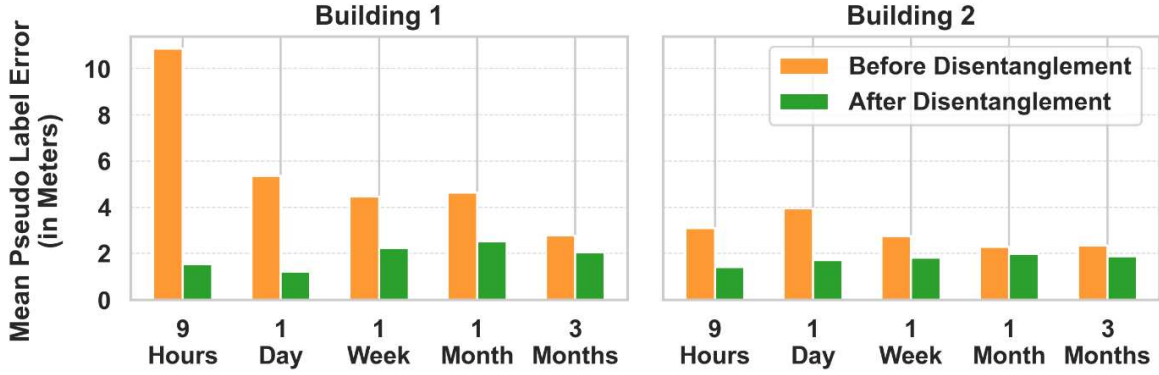


Figure 105: Mean pseudo-label error before and after disentanglement across time in two buildings.

11.6.4. ABLATION STUDY: IMPACT OF DEVICE ONBOARDING ORDER

To evaluate robustness to the order of device introduction over time, we randomly shuffled the sequence of introducing new devices, to eliminate any bias in our results. As shown in Figure 106, across 5 different device orderings over time, DAILOC consistently achieves low localization error, demonstrating resilience to variability in heterogeneous device arrival. In contrast, state-of-the-art methods such as DARE and FEDHIL show moderate sensitivity, while FIDORA exhibits the highest degradation due to its inability to preserve stability across sequential domains, leading to catastrophic forgetting when devices arrive unpredictably.

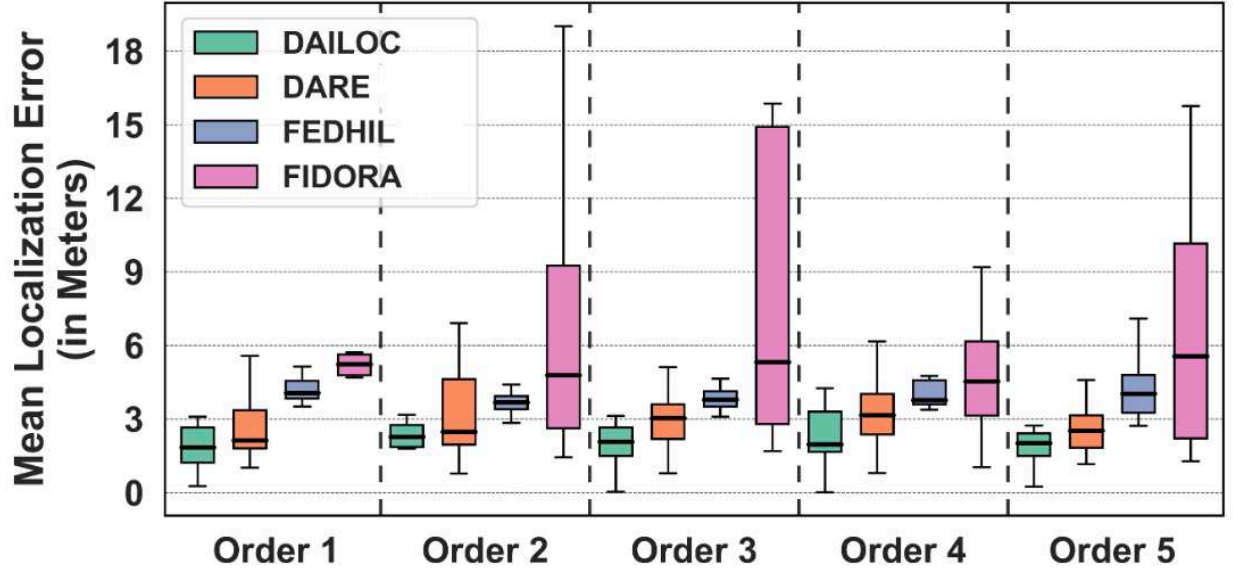


Figure 106: Mean localization error (in meters) across different heterogeneous device introduction orderings over time

11.7. CONCLUSION

In this chapter, we introduced DAILOC, a domain-incremental learning framework for robust Wi-Fi fingerprinting-based indoor localization under real-world conditions. By disentangling domain-specific and location-relevant features using a ML-VAE and leveraging domain-aware reparameterization with memory-guided class latent alignment, DAILOC effectively adapts to both temporal/environmental and device-induced shifts. Evaluations across multiple buildings, smartphones, and deployment stages show that DAILOC consistently outperforms the state-of-the-art, achieving $2.74\times$ lower average error and up to $4.6\times$ lower worst-case error.

CHAPTER 12

GATE: GRAPH ATTENTION NEURAL NETWORKS WITH REAL-TIME EDGE CONSTRUCTION FOR ROBUST INDOOR LOCALIZATION USING MOBILE EMBEDDED DEVICES

Accurate indoor localization is crucial for enabling spatial context in smart environments and navigation systems. Wi-Fi Received Signal Strength (RSS) fingerprinting is a widely used indoor localization approach due to its compatibility with mobile embedded devices. Deep Learning (DL) models improve accuracy in localization tasks by learning RSS variations across locations, but they assume fingerprint vectors exist in a Euclidean space, failing to incorporate spatial relationships and the non-uniform distribution of real-world RSS noise. This results in poor generalization across heterogeneous mobile devices, where variations in hardware and signal processing distort RSS readings. Graph Neural Networks (GNNs) can improve upon conventional DL models by encoding indoor locations as nodes and modeling their spatial and signal relationships as edges. However, GNNs struggle with non-Euclidean noise distributions and suffer from the GNN blind spot problem, leading to degraded accuracy in environments with dense access points (APs). To address these challenges, we propose GATE, a novel framework that constructs an adaptive graph representation of fingerprint vectors while preserving an indoor state-space topology, modeling the non-Euclidean structure of RSS noise to mitigate environmental noise and address device heterogeneity. GATE introduces 1) a novel Attention Hyperspace Vector (AHV) for enhanced message passing, 2) a novel Multi-Dimensional Hyperspace Vector (MDHV) to mitigate the GNN blind spot, and 3) an new Real-Time Edge Construction (RTEC) approach for dynamic graph adaptation. Extensive real-world evaluations across multiple indoor spaces with varying path lengths, AP densities, and heterogeneous devices demonstrate that GATE achieves

1.6× to 4.72× lower mean localization errors and 1.85× to 4.57× lower worst-case errors compared to state-of-the-art indoor localization frameworks.

12.1. INTRODUCTION

Indoor localization has emerged as a critical application in recent years, driving context-aware advancements in smart environments, including spatial robotics, location tracking in augmented/virtual reality (AR/VR), and precise emergency response systems [64]. The first documented indoor localization framework, “The Active Badge Location System”, was proposed in 1992 [224]. This system relied on infrared (IR) signals, where stationary IR receivers detected periodic IR pulses emitted by wearable badges. However, due to the inability of IR signals to penetrate walls, the system was limited to line-of-sight localization, making it impractical for large-scale deployments. As indoor environments have grown increasingly complex, researchers have turned to artificial intelligence (AI) to improve localization accuracy [64], [172]. This shift has driven major investments from industry leaders such as Apple, Google, Microsoft, HPE, Quuppa, and Zebra, driving innovations in AI-powered localization systems [34]. As a result, the global indoor localization market was valued at \$11.9 billion (USD) in 2024, with projections indicating sustained growth fueled by the increasing adoption of intelligent localization technologies [225].

Modern indoor localization primarily relies on data-driven approaches that leverage wireless radio frequency (RF) signal measurements from established protocols such as Wi-Fi, Bluetooth, Ultra-Wideband (UWB), and cellular 4G/5G networks [64], [172], [34]. Among these, Wi-Fi-based localization has gained significant traction due to its widespread availability and seamless integration with mobile devices [226]. Wi-Fi-based localization estimates a device’s location by

measuring Received Signal Strength (RSS) values from Wi-Fi routers or Access Points (APs) that are widely deployed in most indoor spaces today [146]. RSS values, measured in decibels relative to a milliwatt (dBm), typically range from -100 dBm (indicating a weak or unpredictable signal) to 0 dBm (indicating a strong received signal). By analyzing RSS values from multiple APs, a mobile device's location can be estimated. As the device moves, RSS variations across APs enable real-time tracking, making Wi-Fi-based localization a practical solution for many applications. Despite its advantages, Wi-Fi RSS-based localization faces many challenges due to environmental interference and device heterogeneity, both of which degrade localization accuracy [146], [227]. Signal absorption, reflection, and diffraction caused by walls, furniture, and electronic devices introduce unpredictable variations in RSS values. Additionally, human movement, multipath propagation, and shadowing effects further contribute to RSS variations. Device heterogeneity exacerbates these challenges, as different mobile device manufacturers implement proprietary Wi-Fi chipsets, antenna placements, and signal processing algorithms, leading to inconsistent RSS measurements—even for devices at the same location. Prior studies have identified hardware-induced heterogeneity, caused by variations in antennas and processing units (CPU, GPU, NPU, FPGA), and software-induced heterogeneity, where vendor-specific firmware optimizations (e.g., noise filtering) alter raw RSS readings [227], [228]. Collectively, these factors significantly impact the reliability of Wi-Fi-based localization, necessitating robust techniques to mitigate environmental noise and heterogeneity-induced variations.

To address these challenges, researchers have explored various algorithms for localization with wireless signals, such as time difference of arrival (TDoA) and angle of arrival (AoA), with fingerprinting-based localization emerging as the dominant approach for indoor spaces [34], [146], [228]. The fingerprinting method constructs RSS fingerprint vectors, which are feature

representations of RSS values received from multiple APs at a given location. These fingerprints are collected at multiple locations (henceforth referred to as Reference Points (RPs)) across an indoor space, creating a structured database which enables real-time tracking. In general, the fingerprinting-based localization process consists of two key phases: 1) Offline Phase: where RSS fingerprint vectors are collected at all RPs and stored in a database, and 2) Online Phase: where a mobile device at an unknown location captures a new fingerprint vector and compares it against the stored database to determine the closest matching RP.

Despite its widespread adoption, fingerprinting-based localization remains highly sensitive to environmental noise and device heterogeneity, both of which cause substantial variations in RSS values, leading to diminished localization accuracy [136]. Simple similarity-matching approaches with database entries often yield poor results in dynamic indoor environments, which has prompted researchers to explore Machine Learning (ML) based solutions. Models such as K-Nearest Neighbors (KNN) [229], Support Vector Machines (SVM) [230], Gaussian Process Classification (GPC) [17], Deep Neural Networks (DNNs) [42], Convolutional Neural Networks (CNNs) [6], and Attention-based models [45] have shown significant improvements by learning intricate RSS patterns across RPs (as will be discussed in Section 12.3). However, while these models can reduce the impact of environmental noise, they fail to capture spatial relationships between RPs (the topology of the indoor path) and struggle with device heterogeneity induced errors. A fundamental limitation arises from the non-Euclidean nature of Wi-Fi fingerprinting, where noise distribution varies irregularly across the fingerprint vector due to environmental interference and device-specific variations (as will be discussed in Section 12.4). Yet, state-of-the-art ML models for indoor localization assume a Euclidean data structure, treating noise as uniformly distributed

across the fingerprint vector [231]. This incorrect assumption leads to degraded localization performance, particularly in large and heterogeneous environments.

To overcome the limitations of traditional ML approaches, Graph Neural Networks (GNNs) have emerged as a promising solution by integrating graph theory within the ML paradigm, allowing these models to process fingerprint vectors while preserving the topology of the indoor space [232]. Unlike conventional ML models which treat fingerprint vectors as independent feature sets, GNNs explicitly model spatial relationships between RPs by representing the indoor space as a graph. In this representation, each RP serves as a node, embedded with its corresponding fingerprint vector, while edges between nodes are constructed based on spatial proximity and RSS similarity [231], [232]. This structure enables GNNs to learn spatial dependencies instead of treating RPs as isolated points, as discussed later in Section 12.4. As a result, GNNs significantly outperform traditional ML models, particularly in environments where device heterogeneity and environmental noise introduce RSS variations that degrade the performance of conventional approaches [232].

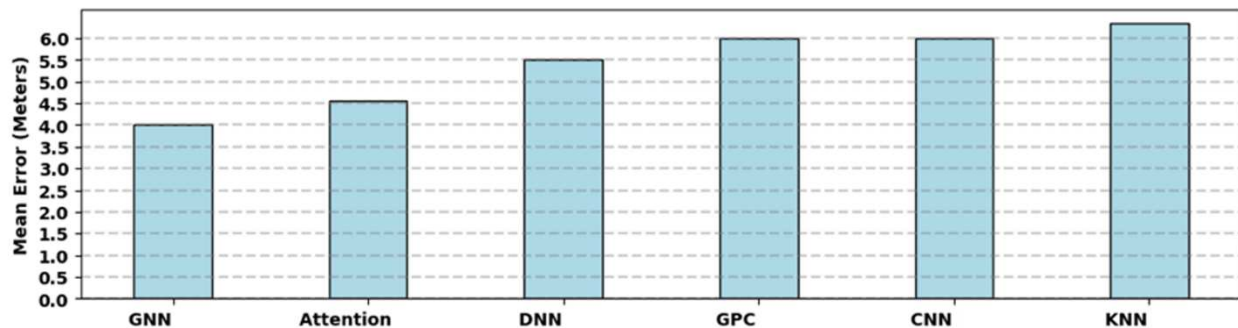


Figure 107: Localization performance comparison of GNN [233] against state-of-the-art ML methods for a 60-meter-long indoor environment. ML models evaluated include an Attention-based Neural Network [45], a fully connected Deep Neural Network (DNN) [42], Gaussian Process Classification (GPC) [17], a Convolutional Neural Network (CNN) [6], and K-Nearest Neighbor (KNN) [229].

To quantify the effectiveness of GNNs for indoor localization, we conducted real-world experiments along a 60-meter-long indoor path in a building environment, where RPs were spaced 1 meter apart, and each RP observed up to 183 APs (thus a fingerprint vector at each RP has a maximum dimension of 183). To assess device heterogeneity, RSS fingerprint data was collected using seven different mobile devices (smartphones). Further details on dataset specifications, device configurations, and building floorplans are provided in Section 12.6. Figure 107 presents a comparative analysis of a traditional GNN model-based solution [233] against other ML models used in state-of-the-art indoor localization solutions. Our results demonstrate that the GNN achieves the lowest localization error, outperforming KNN [229] and Attention-based models [45]— both widely used in prior works (and discussed further in Section 12.3). Despite their advantages, our analysis indicates that GNNs suffer from scalability issues in large indoor environments with many APs, primarily due to the "GNN blind spot problem", where conventional GNNs struggle to capture long-range dependencies in high-dimensional graphs (in the presence of high AP density per RP) and fail to model the non-Euclidean distribution of noise in RSS fingerprints [234]. Since GNNs rely on message passing to propagate node information through edges, they inherently assume a Euclidean noise structure, which unfortunately degrades performance in noisy, real-world settings. While message passing enables nodes to aggregate information from edges, it loses the relative influence of each edge on the node, making the learned representations noisy and less discriminative. This effect is further exacerbated in environments with high AP densities, where multiple APs contribute inconsistent RSS values, amplifying RSS variations and reducing localization accuracy.

12.2. CONTRIBUTIONS

To collectively address environmental noise, device heterogeneity, GNN blind spots, and non-Euclidean data modeling, in this work we introduce GATE (Graph Attention Neural Networks with Real-Time Edge Construction)—a novel GNN-based framework for robust indoor localization. Our key contributions are:

- We introduce the Attention Hyperspace Vector (AHV), which improves message passing by preserving the non-Euclidean noise structure of fingerprint vectors, ensuring more robust edge influence modeling.
- We propose the Multi-Dimensional Hyperspace Vector (MDHV), which captures an indoor topology while incorporating edge influence to mitigate the GNN blind spot problem.
- We develop a Real-Time Edge Construction (RTEC) policy, which dynamically refines graph connectivity during inference, enhancing resilience to environmental noise and device heterogeneity.
- We conduct extensive real-world experiments across diverse buildings with varying path lengths, AP densities, and multiple heterogeneous devices to evaluate the GATE framework’s robustness.
- We benchmark GATE’s performance against state-of-the-art GNN-based indoor localization methods, demonstrating its effectiveness in the presence of real-world RSS variations.

The remainder of this chapter is structured as follows: Section 12.3 reviews related works in the domain of indoor localization, highlighting key advancements and approaches that have been proposed to address localization challenges. Section 12.4 delves into the nature of RSS fingerprint vectors, examining their inherent structure and discussing methods to represent these vectors as a graph. Section 12.5 introduces the GATE framework, detailing its architecture, components, and

operational workflow. Section 12.6 presents the experimental setup and is followed by a comprehensive evaluation of GATE’s performance against state-of-the-art indoor localization solutions. Lastly, Section 12.7 summarizes our key findings.

12.3. RELATED WORK

With the adoption of ML in indoor localization in recent years, a wide range of approaches have been proposed to make RSS fingerprinting more viable in real-world deployments. Classical ML algorithms such as K-Nearest Neighbors (KNN) [229], Support Vector Machines (SVM) [230], Gaussian Process Classification (GPC) [17], Decision Trees (DT) [235], and Non-Negative Matrix Factorization (NMF) [158] showed improvements over basic similarity-matching approaches by learning patterns in fingerprint vectors across RPs. These models served as an early validation of the feasibility of ML in fingerprinting-based indoor localization. However, they remained highly sensitive to variations in RSS due to environmental noise and device heterogeneity, as they lacked the capacity to differentiate between meaningful RSS patterns and noise artifacts. This limitation prompted a transition toward deeper ML models such as Deep Neural Networks (DNNs) [42] and Convolutional Neural Networks (CNNs) [6], which improved performance by extracting higher-level representations from fingerprint vectors. The deeper models are capable of separating RSS patterns from small-scale random noise by focusing on higher-order feature representations. Yet, in real-world environments, RSS fluctuations are not only random but also follow non-uniform noise distributions (non-Euclidean structure) that these models are unable to efficiently generalize over [231].

To better capture such variability, researchers began leveraging data augmentation techniques during training to simulate real-world noise conditions. These included synthetic noise injection

strategies using architectures such as autoencoders and generative adversarial networks (GANs). Notable frameworks like SANGRIA [82], WiDeep [17], and DataLoc [178] explored several data augmentation methods to generate augmented RSS fingerprints, effectively mimicking environmental and heterogeneity-induced noise. Simultaneously, approaches like STONE [32], and STELLAR [134] incorporated contrastive learning, which relies on augmentation to learn invariant representations by contrasting noisy and clean RSS samples. These methods enriched the training distribution and offered improved resilience to moderate noise. However, despite their successes, these models continued to treat fingerprint vectors as existing in a Euclidean space, assuming noise to be uniformly distributed across all APs. None of these methods model the spatial relationships between RPs, i.e., the indoor path topology, which is critical for understanding how RSS values change with user movement and the physical layout of the indoor environment. Without incorporating this topological context, existing solutions that use deep ML models lack the spatial awareness required for high accuracy indoor localization.

To address the structural limitations of conventional deep ML models, Graph Neural Networks (GNNs) have recently emerged as a promising paradigm for fingerprinting-based indoor localization by modeling fingerprint vectors as graphs [233]. In this formulation, RPs are represented as graph nodes, and edge connections capture spatial relationships or RSS similarities between nodes. This allows GNNs to preserve the topological structure of the indoor environment, which traditional deep ML models lack. However, GNNs are still relatively new in this domain, with only a few published works. One of the earliest attempts was GNN-LOC proposed in 2021 [233]. Although this approach leveraged graph-based modeling to encode indoor topology, it lacked message passing, and therefore failed to model the influence of the edges—particularly the interference effects caused by overlapping APs in nearby RPs. This limitation led to the

development of extensions like GNN-ED [236] and GCN-ED [237], which continued to use the Euclidean distance for edge construction—typically by thresholding the average pairwise Euclidean distance across fingerprint vectors—but added message passing layers to enable feature aggregation across connected nodes. GCN-ED additionally interpreted the graph as an image and applied convolutional filters to learn spatial patterns. Building on this, GNN-KNN [238] and GCN-KNN [239] used KNN for edge construction, where edges were formed based on RSS similarity. These models improved performance by incorporating message passing to learn topological dependencies and demonstrated better resilience to environmental noise. However, they still operated under the assumption of a Euclidean data structure and failed to account for non-uniform noise introduced by device heterogeneity. To further improve learning under such variability, attention mechanisms were integrated into the GNN framework. In particular, GCLoc [240] and GraphLoc [244], which are variants of Graph Attention (GAT) networks allowed each node to assign learned attention weights to its edges, effectively prioritizing informative connections and reducing the influence of unreliable or noisy nodes. However, even GCLoc and GraphLoc remains limited by its reliance on Euclidean assumptions and its vulnerability in high-density AP environments, where modeling becomes increasingly complex.

In summary, despite recent advances, GNN-based indoor localization solutions continue to underperform in environments with high AP densities (experimental evidence across varying AP densities is presented in Section 12.6.5). In such settings, the graph structure becomes increasingly noisy, leading to incorrect or redundant edge connections that disrupt spatial dependencies. We term this as the "GNN blind spot problem", which arises when message passing is insufficient to transmit information across nodes with high AP density, resulting in feature dilution. While the specific term "GNN blind spot" is not universally adopted, the phenomenon has been recognized

in GNN literature as oversquashing [241], over-smoothing [242], and neighborhood dilution [243]. Together, these limitations highlight the need for a more advanced framework that overcomes three fundamental challenges: 1) overcoming both environmental noise and device heterogeneity by preserving the inherently non-Euclidean structure of fingerprint vectors, 2) maintaining indoor topology within the graph structure to ensure spatial context is retained, and 3) addressing the GNN blind spot by enabling robust message passing even in high-density AP environments.

12.4. REPRESENTING FINGERPRINT VECTORS AS A GRAPH FOR INDOOR LOCALIZATION

In this section, we first analyze and discuss the structural properties of fingerprint vectors and how they affect the learning dynamics of conventional ML models. Next, we explore graph-based representations of fingerprint vectors, explaining how spatial and signal-based relationships can be effectively encoded to enhance localization accuracy. Lastly, we examine existing message passing algorithms in GNNs, discussing their implementations and how they contribute to the GNN blind spot challenge. The insights from this analysis inform the development of GATE, ensuring that the framework preserves the structural properties of fingerprint vectors while addressing the limitations of existing GNN based solutions.

12.4.1. STRUCTURAL ANALYSIS OF WI-FI RSS FINGERPRINT VECTORS: EUCLIDEAN VS. NON-EUCLIDEAN REPRESENTATION

Determining whether a dataset exhibits a Euclidean or non-Euclidean structure is critical when selecting an appropriate learning algorithm. In a Euclidean paradigm, data points are assumed to lie in a space where distances can be measured using Euclidean distance (ED)—the straight-line distance between two points. This structure imposes fixed, uniform relationships between features,

implying that variations in RSS values across different APs are evenly distributed, with a constant noise offset (δ) applied uniformly across all dimensions of the fingerprint vector. Most ML models are inherently designed to operate within such Euclidean spaces. Each RSS value in the fingerprint vector is treated as an independent feature, and these models implicitly assume that noise—whether from environmental interference or device heterogeneity—is uniformly distributed across all APs, as shown in Figure 108 (left). However, real-world fingerprint vectors violate these assumptions. The variations introduced by environmental and device-specific factors are inherently non-uniform, meaning that some APs are affected more than others depending on physical layout, environmental noise, and device heterogeneity. This assumption degrades the generalizability of traditional ML models, limiting their effectiveness in real-world indoor localization. For example, if a fingerprint vector is denoted as F , as shown in Equation 69, Euclidean models assume a constant offset δ across all APs, resulting in a transformed vector $F_{Euclidean}$, shown in Equation 70.

$$F = [RSS_{AP:0}, RSS_{AP:1}, \dots, RSS_{AP:N}] \quad (69)$$

$$F_{Euclidean} = [(\delta, RSS_{AP:0}), (\delta, RSS_{AP:1}), \dots, (\delta, RSS_{AP:N})] \quad (70)$$

In reality, the δ introduced at each AP varies in a non-uniform manner. Therefore, the true F follows a non-Euclidean structure, where each AP-specific offset δ_i differs, as expressed in Equation 71.

$$F_{Non-Euclidean} = [(\delta_0, RSS_{AP:0}), (\delta_1, RSS_{AP:1}), \dots, (\delta_N, RSS_{AP:N})] \quad (71)$$

Figure 108 (right) illustrates this effect, showing how RSS values exhibit spatially inconsistent δ_i , disrupting the structured relationships assumed by Euclidean models. Hence, to accurately process F a more sophisticated learning approach is required—one that can operate effectively in non-Euclidean spaces.

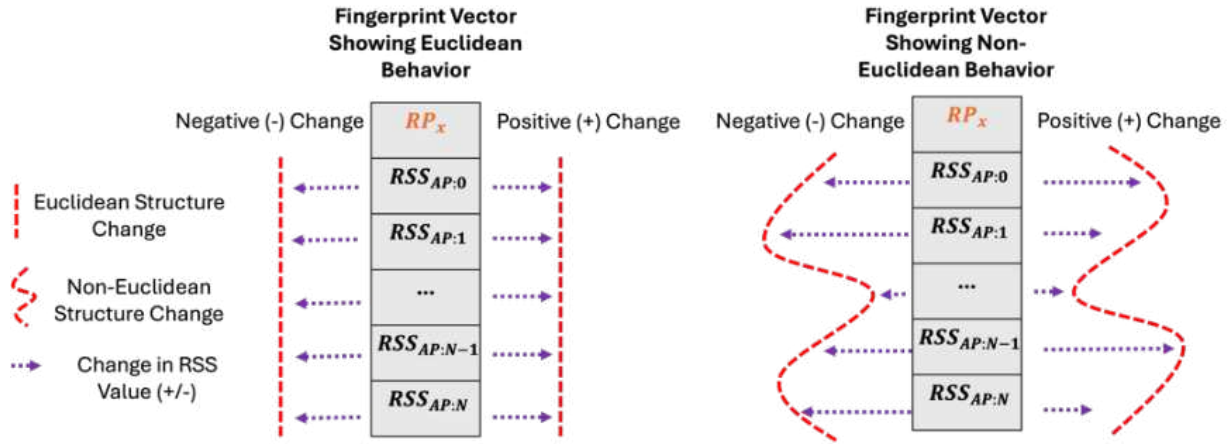


Figure 108: Fingerprint vector with Euclidean noise assumption (left) vs. real-world non-Euclidean structure (right).

12.4.2. REPRESENTING FINGERPRINT VECTORS AS A GRAPH

To effectively leverage GNNs for indoor localization, it is essential to construct a graph representation of fingerprint vectors F that captures both the topology of the indoor environment—i.e., how RPs are spatially structured along the indoor path—and the inherent non-Euclidean relationships among them. Let F_i denote F at RP i . Unlike traditional ML models that treat F_i as an independent feature vector in Euclidean space, a graph-based approach explicitly models the spatial dependencies between RPs. This enables the model to extract localization cues not just from direct fingerprint similarities, but also from the spatial context—crucial for generalization in environments affected by irregular δ_i . In this representation, each RP is mapped as a node in the graph, with its associated F serving as the node feature, as illustrated in Figure 109 (left). Edges

between nodes encode spatial and signal-based relationships, and their structure should ideally reflect the physical layout and signal dynamics of the environment. Formally, this graph (G) can be defined as:

$$G = (V, E, F) \quad (71)$$

Here, V denotes the set of nodes, where each node $v_i \in V$ corresponds to an RP with its associated fingerprint vector $F_i \in R^N$, where N represents the number of available APs. The set of edges $E \in \{(v_i, v_j)\}$ captures relationships between RPs and can be defined based on spatial proximity or RSS similarity. The effectiveness of this graph representation hinges on the quality of the edge construction strategy. If the edges do not accurately reflect the underlying topology of the indoor environment, the graph structure fails to model meaningful spatial dependencies, ultimately degrading localization performance. Many existing GNN-based approaches employ static graph construction methods, such as KNN or ED-based RSS similarity, discussed next in Section 12.4.3. These methods often overlook the non-uniformity of RSS noise and treat all AP measurements as equally reliable, which can lead to incorrect node associations, linking RPs with similar RSS values but distant physical locations. Such erroneous connections distort the topology of the graph and introduce noise into the message passing process. To enable effective learning, the graph must accurately preserve local spatial relationships between nearby RPs while also maintaining enough global structure to model long-range dependencies within the indoor environment. However, achieving this balance is challenging due to the irregular nature of RSS data, and remains a key limitation in current GNN-based architectures used for indoor localization.

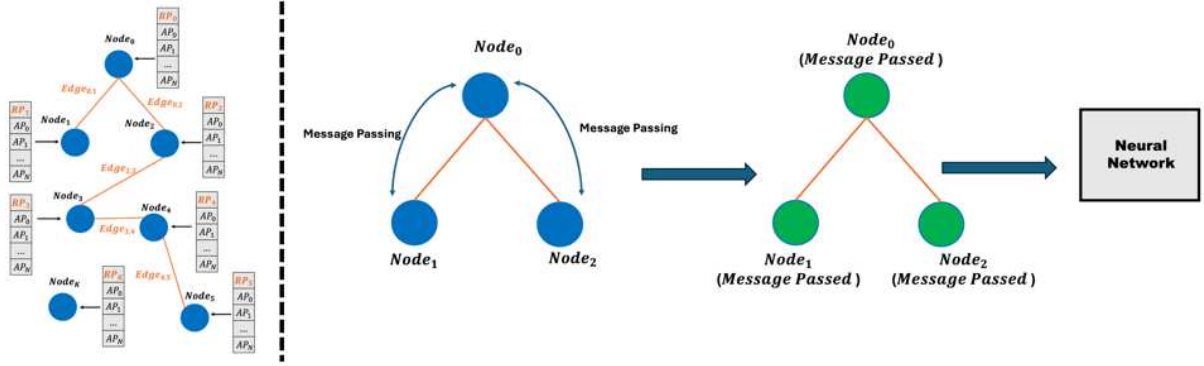


Figure 109: Graph representation of fingerprint vectors (left) and message passing between nodes (right).

Although constructing an appropriate graph structure is a critical step, it alone is insufficient. Once the graph is defined, GNNs rely on message passing to exchange information across nodes, enabling the model to refine each RP's representation based on its neighbors. However, in large-scale indoor environments with dense AP deployments, conventional message passing strategies become less effective, which we define as the GNN blind spot problem. In the next section, we analyze the working of existing message passing algorithms and motivate our design choices for GATE.

12.4.3. EDGE CONSTRUCTION AND MESSAGE PASSING FOR FINGERPRINT VECTORS

Once fingerprint vectors F are represented as a graph, the accuracy and efficiency of indoor localization heavily depend on how edges E are constructed and how information is propagated between nodes V through message passing. The E in a GNN define how spatial and signal-based dependencies are captured, while message passing dictates how information flows between RPs, influencing the model's ability to infer relationships between V . Incorrect E construction can

distort the non-Euclidean relationships in F , leading to incorrect feature propagation and degraded localization accuracy. A few E construction and message passing strategies have been explored for GNN-based localization, each with distinct advantages and limitations. We analyze the three most widely used approaches next.

12.4.4. EUCLIDEAN DISTANCE-BASED MESSAGE PASSING

In Euclidean Distance (ED) based edge construction, nodes V are connected based on the average pairwise ED between their respective F . The idea is that nodes with low ED values are assumed to be more similar in signal space and are therefore likely to be spatially or topologically related. A threshold $\emptyset_{ED}$ is set to filter connections, ensuring that only nodes with sufficiently similar RSS patterns are linked. The ED between two nodes (v_i, v_j) is computed as:

$$ED(v_i, v_j) = \frac{1}{N} \sum_{k=1}^N |F_i^k - F_j^k|^2 \quad (72)$$

where F_i^k denotes the RSS value from the k^{th} AP in the fingerprint vector F_i and N is the total number of APs. Once the ED score is calculated, an edge is formed between (v_i, v_j) only if their ED falls below $\emptyset_{ED}$ as shown below:

$$E = \{ (v_i, v_j) \mid ED(v_i, v_j) \leq \emptyset_{ED} \} \quad (73)$$

After edge construction, feature aggregation is performed using weighted message passing scheme. Each node updates its representation by averaging the features of its neighbors, scaled by learned weights $\alpha_{i,j}$:

$$F'_i = \sigma(W * \sum_{j \in N(v_i)} (\alpha_{i,j} * F_j)) \quad (74)$$

Here W is the trainable weight matrix and σ is a non-linear activation function such as *RELU*. Thus, ED-based edge construction connects nodes with similar RSS values, but this approach suffers from key limitations in real-world indoor localization. Due to environmental noise and device heterogeneity, two distant RPs may occasionally share similar fingerprints. As a result, this method can produce incorrect edges—either missing valid neighbors or including unrelated nodes—leading to noisy message passing and reduced localization accuracy.

12.4.5. K-NEAREST NEIGHBOR (KNN)-BASED MESSAGE PASSING

In KNN based edge construction, each node is connected to its K most similar neighbors, typically using ED as the similarity metric. While the same distance function is used as in threshold-based ED construction (see Eq. (5)), the key distinction lies in how edges are formed. Rather than applying a global threshold (Eq. (6)), KNN uses a local ranking strategy, forming edges to the K nodes with the lowest ED scores relative to each node. This ensures that each node maintains exactly K connections, promoting stable graph connectivity even in the presence of environmental noise or signal sparsity. Formally, an edge $(v_i, v_j) \in E$ is created only if v_j lies among the top K nearest neighbors of v_i based on the ED ranking:

$$E = \{ (v_i, v_j) \mid v_j \in k_{v_i}(ED(v_i, v_j)) \} \quad (75)$$

Once edges are formed, message passing proceeds similarly to the formulation shown in Equation 75. KNN-based construction offers two notable advantages over global threshold-based ED: 1) it guarantees that each node has a consistent number of neighbors, mitigating the risk of

disconnected subgraphs; and 2) it adapts locally to fingerprint distributions, making it more robust to noise variations across the environment. However, it still shares a major limitation with ED-based methods—both rely on Euclidean similarity, which does not inherently account for the non-Euclidean structure of fingerprint vectors. As a result, KNN may still form spurious edges between signal-similar but spatially distant nodes.

12.4.6. ATTENTION-BASED MESSAGE PASSING

Unlike KNN-based edge construction, which enforces a fixed number of neighbors, or ED-based methods that rely on a global threshold, attention-based message passing learns the importance of each neighbor during training. Instead of using static edge weights, the attention mechanism assigns a learnable importance score to each edge based on the attention similarity between the node features. Given two node embeddings F_i and F_j , the attention coefficient $\alpha_{i,j}$ is computed as:

$$\alpha_{i,j} = \frac{\text{EXP}(\text{LeakyRELU}(a^T [W * F_i \parallel W * F_j]))}{\sum_{K \in N(v_i)} \text{EXP}(\text{LeakyRELU}(a^T [W * F_i \parallel W * F_j]))} \quad (76)$$

where W is a learnable weight matrix, a^T is the learnable attention vector, and $\parallel$ denotes concatenation. Once the attention coefficients are computed, message passing proceeds using the same weighted aggregation described in Equation 75, where neighbors are now weighted by $\alpha_{i,j}$. This allows the model to adaptively prioritize informative nodes and ignore noisy or less useful ones. However, despite their advantages, attention mechanisms still depend on the initial Euclidean structure, which may introduce spurious connections. In environments with dense AP

distributions, they remain vulnerable to the GNN blind spot, where long-range dependencies are lost, and message passing becomes ineffective.

12.5. GATE FRAMEWORK FOR ROBUST INDOOR LOCALIZATION

GATE introduces a novel graph-based learning framework for Wi-Fi RSS fingerprinting that preserves both the indoor topology and the non-Euclidean structure of fingerprint vectors. Unlike traditional ML models that treat fingerprint vectors as independent features in Euclidean space, GATE first establishes edges based on the indoor topology, ensuring that spatial relationships between RPs are preserved. It then assigns attention scores to each edge, based on the similarity between node embeddings and also models the graph to preserve the non-Euclidean structure of fingerprint vectors. This combined strategy enables GATE to adaptively capture both indoor topology and signal-based dependencies, addressing key challenges such as environmental noise, device heterogeneity, and the GNN blind spot problem.

GATE operates in two key phases: an offline training phase and an online inference phase, as illustrated in Figure 110. During the offline phase, fingerprint vectors are collected at multiple RPs across the building. Each RP is treated as a node in a graph, where the total number of nodes corresponds to the number of RPs in the training set. To establish edges between nodes, GATE employs a hybrid approach that considers both indoor topology and signal-based attention scores. Edges are first formed based on indoor topology, and attention scores are then assigned to quantify the relative importance of each edge. These scores are computed using a scalar dot product between node embeddings, as detailed in Section 12.5.2. The number of neighbors per node is treated as a tunable hyperparameter, whose influence on localization performance is examined in Section 12.6.2. Nodes with higher attention scores are emphasized during training, allowing the model to

strengthen meaningful spatial dependencies while suppressing weak or noisy associations that could hinder accuracy.

Once the graph is constructed, GATE computes a Multi-Dimensional Hyperspace Vector (MDHV) for each node, which serves as the primary feature representation for training, as shown in Figure 110. The MDHV consists of three components: 1) the fingerprint vector of the node, representing raw RSS values, 2) the attention-based message passing (MSG) vector, which aggregates information from connected nodes based on their edge weights, and 3) the Attention Hyperspace Vector (AHV), a multi-dimensional tensor encoding the feature-wise influence of each neighboring node. While the MSG vector captures the overall influence of connected nodes, the AHV preserves the non-Euclidean structure by maintaining per-feature contributions, thereby accounting for the non-uniform impact of environmental noise and device heterogeneity. Together, these components enable the MDHV to encode both the influence of each edge and the fine-grained contribution of each neighbor's features, preserving spatial integrity and signal variability in a structured form. This representation enhances robustness against non-uniform distortions and directly addresses the GNN blind spot problem by ensuring that feature-level distinctions are retained even under dense AP conditions. Once the MDHVs are computed for all nodes, they are passed into a custom-designed Graph Convolutional Network (GCN) for training. We select GCNs because the MDHV structure resembles image-like spatial feature maps, making convolutional layers particularly well-suited to extract hierarchical patterns and spatial relationships. The GCN architecture, as shown in Figure 110, consists of two graph convolutional layers followed by a fully connected classification layer. The first layer captures high-order dependencies across the graph, while the second compresses the feature space to retain only the most discriminative attributes before final location prediction.

In the online phase, when a mobile device captures an RSS fingerprint at an unknown location, it is temporarily added to the graph as a new node without altering the trained GCN. GATE applies Real-Time Edge Construction (RTEC), described in Section 12.5.5, to compute attention scores between the new node and existing nodes, assigning edges to the most relevant ones. The number of edges is kept fixed to match the training configuration and maintain dimensional consistency. The MDHV for the new node is then computed using the same approach as in training, ensuring compatibility with the learned graph structure and allowing the GCN to accurately predict the most likely RP.

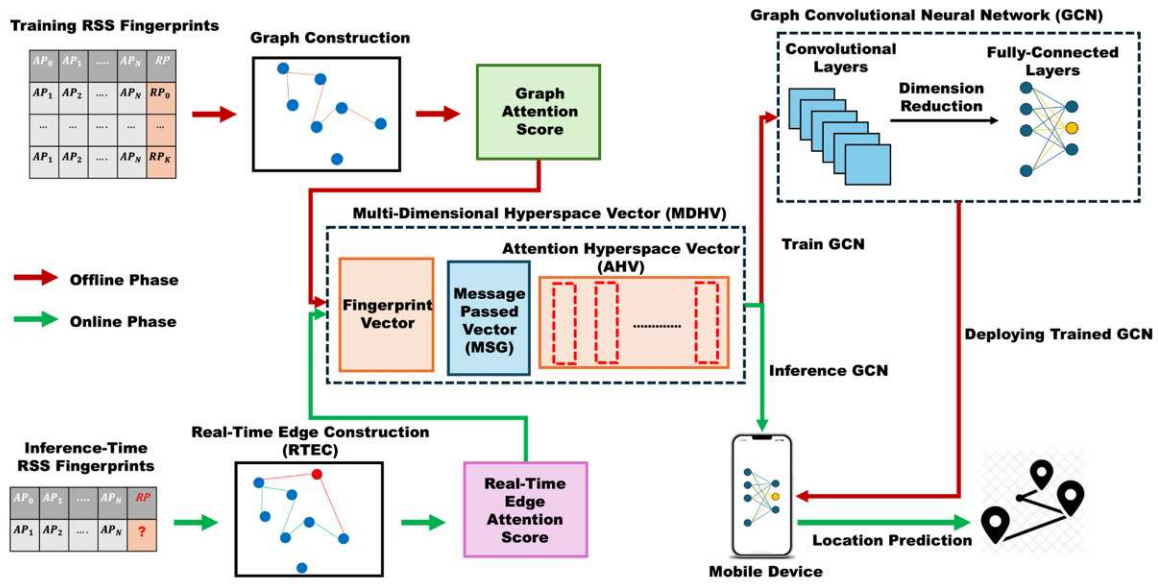


Figure 110: Overview of the GATE framework during offline training and online inference phases.

12.5.1. GATE ATTENTION SCORE MECHANISM

The attention score mechanism in GATE is designed to quantify the similarity between fingerprint vectors and determine the strength of connections between nodes in the graph. Given

two RSS fingerprint vectors, F_i and F_j , corresponding to two RPs, the attention score is computed using the scalar dot product attention. This is defined as:

$$a_{i,j} = \frac{F_i \cdot F_j}{||F_i|| \cdot ||F_j||} \quad (77)$$

where $a_{i,j}$, represents the attention score between node i and node j , and the dot product captures the alignment between the two fingerprint vectors. The denominator normalizes the score, ensuring that the attention values remain bounded and comparable across different RPs. This formulation ensures that fingerprint vectors with similar RSS distributions receive higher attention scores, effectively modeling the spatial and signal-based relationships between locations. Once attention scores are computed for all connected nodes, they are used as weights in a weighted averaging scheme to perform message passing. The message passing process aggregates feature information from neighboring nodes while ensuring that more relevant nodes contribute more significantly to the target node's representation. The aggregated message MSG_i for node i is computed as:

$$MSG_i = \sum_{j \in N(i)} (A_{i,j} \cdot F_j) \quad (78)$$

where $N(i)$ represents the set of neighboring nodes connected to node i . This message passing step ensures that the spatial dependencies between fingerprint vectors are encoded into the node representation, improving the model's ability to generalize across different environments. The computed message vector MSG_i , along with the original fingerprint vector F_i , is then integrated into the Multi-Dimensional Hyperspace Vector (MDHV) for further computations, as discussed next.

12.5.2. MULTI-DIMENSIONAL HYPERSPACE VECTOR (MDHV) COMPUTATION

The Multi-Dimensional Hyperspace Vector (MDHV) serves as a representation for each node in the GATE framework, ensuring that the spatial and signal-based dependencies within the fingerprint vectors are effectively captured. The MDHV consists of three components: 1) the fingerprint vector F_i , which contains the raw RSS values from multiple APs, 2) the message-passed MSG_i vector, which aggregates information from connected nodes based on attention-weighted averaging, and 3) the Attention Hyperspace Vector (AHV), a multi-dimensional tensor encoding the feature-wise influence of neighboring nodes. The AHV plays a crucial role in capturing the non-uniform impact of each neighboring node's features on the target node. While traditional attention mechanisms assign a single weight per node, the AHV expands this concept by computing feature-wise attention, ensuring that the contribution of each RSS feature is dynamically adjusted. The AHV is computed as:

$$AHV_{i,j} = \frac{F_i \odot F_j}{||F_i|| \cdot ||F_j||} \quad (79)$$

where $\odot$ denotes element-wise multiplication, ensuring that each feature within F_i is compared individually against F_j , a neighboring node. This formulation ensures that RSS features with higher correlation across nodes receive greater weightage during message passing, while irrelevant or noisy features contribute minimally. The AHV is structured as a multi-dimensional tensor, where each column corresponds to a connected node and each row represents an individual attention score for each RSS feature, with the total dimensionality dictated by the number of connected edges per node. By integrating the fingerprint vector, message-passed vector, and the AHV, the MDHV preserves both the local and global dependencies within the graph while maintaining the

non-Euclidean structure of the fingerprint vectors. This structured representation is then supplied as input to the Graph Convolutional Network (GCN), allowing the model to effectively learn spatial and signal-based patterns crucial for accurate indoor localization.

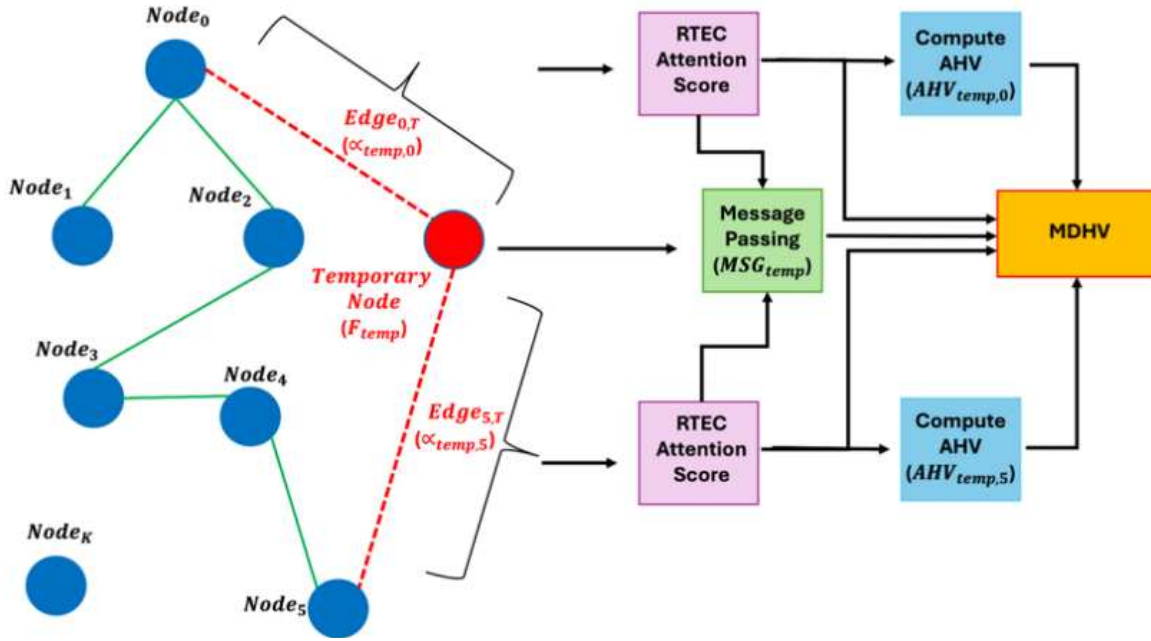


Figure 111: Overview of the RTEC algorithm during the online phase.

12.5.3. REAL-TIME EDGE CONSTRUCTION (RTEC) ALGORITHM

The Real-Time Edge Construction (RTEC) algorithm plays a crucial role in integrating new fingerprint vectors dynamically into the GATE framework during the online phase. Unlike static graph structures used in conventional GNN-based models, RTEC dynamically constructs edges for a temporary node (e.g., red node in Figure 111). When a new RSS fingerprint vector F_{temp} is captured from an unknown location, RTEC treats it as a temporary node in the graph. To establish meaningful connections with existing nodes, RTEC computes attention scores between F_{temp} and the fingerprint vectors of pre-existing RPs using the same scalar dot-product attention mechanism

applied in the offline phase. The attention score between the temporary node and an existing node i is given by:

$$\alpha_{temp,i} = \frac{F_{temp} \cdot F_i}{||F_{temp}|| \cdot ||F_i||} \quad (80)$$

where $\alpha_{temp,i}$ measures the similarity between the newly observed fingerprint and the stored fingerprint of RP i . The number of edges connected to the temporary node is kept fixed, as determined during the offline phase, to maintain dimensional consistency in the graph structure. Once the edges are determined, message passing is performed using attention-weighted averaging, aggregating information from the selected neighbors, as shown in Figure 111. The message-passing output MSG_{temp} is computed as:

$$MSG_{temp} = \sum_{i \in N_{temp}} (\alpha_{temp,i} \cdot F_i) \quad (81)$$

where N_{temp} represents the set of selected neighbors for the temporary node. The MDHV is then constructed using the same methodology as in the offline phase, integrating the fingerprint vector, the message-passed vector, and the AHV, which captures the feature-wise impact of each connected node. Finally, the constructed MDHV is passed into the pre-trained GCN, which predicts the most probable RP associated with the fingerprint vector. RTEC allows GATE to perform real-time localization by dynamically selecting neighbors based on signal similarity, computing attention-aware feature aggregations, and maintaining structural consistency with the trained graph.

12.6. EXPERIMENTAL SETUP

We evaluated the performance of the GATE framework using real-time data collection to closely emulate practical deployment scenarios. Experiments were conducted across five distinct buildings, with floorplans in each differing in the number of RPs, AP densities, and spatial topologies—factors that introduce varying and challenging environmental noise characteristics. To additionally consider the impact of device heterogeneity, we conducted experiments with seven commercially available smartphones with diverse hardware and software configurations. For each RP, up to five fingerprint vectors were collected for training, while one fingerprint vector was reserved for testing. The spatial layouts of the buildings and the specifications of the devices used are illustrated in Figure 112.

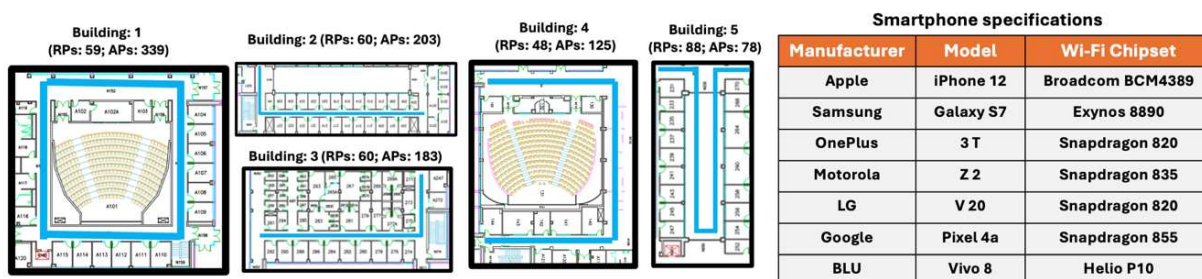


Figure 112: Spatial layouts of the building floorplans (left) and mobile device specifications (right).

We selected five buildings (Building 1–5), each chosen for its unique structural and environmental characteristics. The test paths span various shapes and lengths, ranging from 48 to 88 meters. Building 1 exhibits the highest AP density, with up to 339 visible APs per RP and 59 RPs, resulting in a fingerprint vector of shape (59, 339). It features a mix of wood and concrete infrastructure and is surrounded by computers in large open areas. In contrast, Building 5 has the lowest AP density, with only 78 APs per RP but the highest RP count of 88, forming a fingerprint vector of shape (88, 78). This building contains laboratory equipment with several different

electronic and mechanical devices. Buildings 2 and 3 each have 60 RPs, but differ in AP densities: Building 2 has 203 APs per RP (60, 203), while Building 3 has 183 APs per RP (60, 183). Both buildings are newer constructions with a blend of metallic and wooden structures, containing open spaces, bookshelves, and heavy metallic objects. Building 4 represents the shortest path, with 48 RPs and the second-lowest AP density of 125 (48, 125). It includes multiple small office rooms with a combination of metal, wood, and glass partitions. To ensure consistent data collection across all environments, a fixed granularity of 1 meter was maintained between adjacent RPs. For training, we used the Motorola Z2 device, while all seven smartphones were employed for testing to comprehensively evaluate generalization under device heterogeneity.

The training process for GATE is designed to optimize model efficiency while ensuring high accuracy indoor localization. We configure the GCN with a learning rate of 0.001 and utilize the sparse categorical cross-entropy loss function. The first CONV layer in the GCN is configured with filters equal to the size of the MDHV, applying a kernel size of 1. The second CONV layer reduces the feature dimension to H (GCN compression value), where we explore the impact of varying H values in Section 12.6.2. Both CONV layers employ the RELU activation function, followed by a fully connected layer with neurons set to the number of RP classes for the corresponding building floorplan. The final classification output uses the *Softmax* activation function. The model was trained for 1,000 epochs, with a total of 51,542 trainable parameters, and a compact model size of 604 KB, making it feasible for efficient deployment on resource-limited devices, such as wearables and smartphones. For performance evaluation, we measure localization error using the absolute distance metric. The error for each prediction is computed as the absolute difference between the predicted RP (RP_{Pred}^i) and the ground truth RP (RP_{True}^i) in meters, as shown below:

$$Error_{Device} = \frac{1}{N} \sum_{i=1}^N |RP_{Pred}^i - RP_{True}^i| \quad (82)$$

where N is the total number of RPs, RP_{Pred}^i is the predicted RP class for RP i , and RP_{True}^i is the ground truth RP for RP i . The mean localization error for a given test device ($Error_{Device}$) is calculated as the average of these absolute differences across all RPs within a building floorplan.

12.6.1. SENSITIVITY ANALYSIS: GCN COMPRESSION SIZE AND NUMBER OF EDGES PER NODE

In this section, we show results of explorations on the sensitivity of GATE to two core hyperparameters: the GCN compression size (H) and the number of edges per node (NB), as shown in Figure 113. The compression size H represents the dimensionality reduction between the first and second convolutional (CONV) layers in the GCN and is varied from 0% (no compression) to 90% (retaining only 10% of the original feature dimensions), plotted along the Y-axis. NB defines the percentage of nodes connected to each node based on attention scores, and is varied from 1% (sparse connections) to 100% (fully connected graph), plotted along the X-axis. The Z-axis in each plot shows the mean localization error (in meters) averaged over seven testing devices. This setup allows us to systematically study how GATE responds to changes in feature compression and edge density, and more importantly, where GNN blind spots begin to emerge.

Starting with compression size, we observe that Buildings 1, 2, and 3—each characterized by high AP densities and hence high-dimensional fingerprint vectors—exhibit a relatively flat localization error surface across H values. This indicates that GATE is robust to feature compression in such environments, with only minor improvements around the 40% to 60% compression range. The model can afford to drop some feature dimensions without significant

degradation because the original vectors already capture rich spatial and signal characteristics. In contrast, Buildings 4 and 5, which have lower AP densities and therefore smaller fingerprint vectors, show noticeable increases in error beyond 60% compression. In these environments, overly aggressive compression leads to the loss of key discriminative features within the MDHV, resulting in reduced localization performance. Overall, the aggregated performance across all buildings confirms that GATE remains largely stable with respect to H, with 50% compression yielding the lowest mean errors.

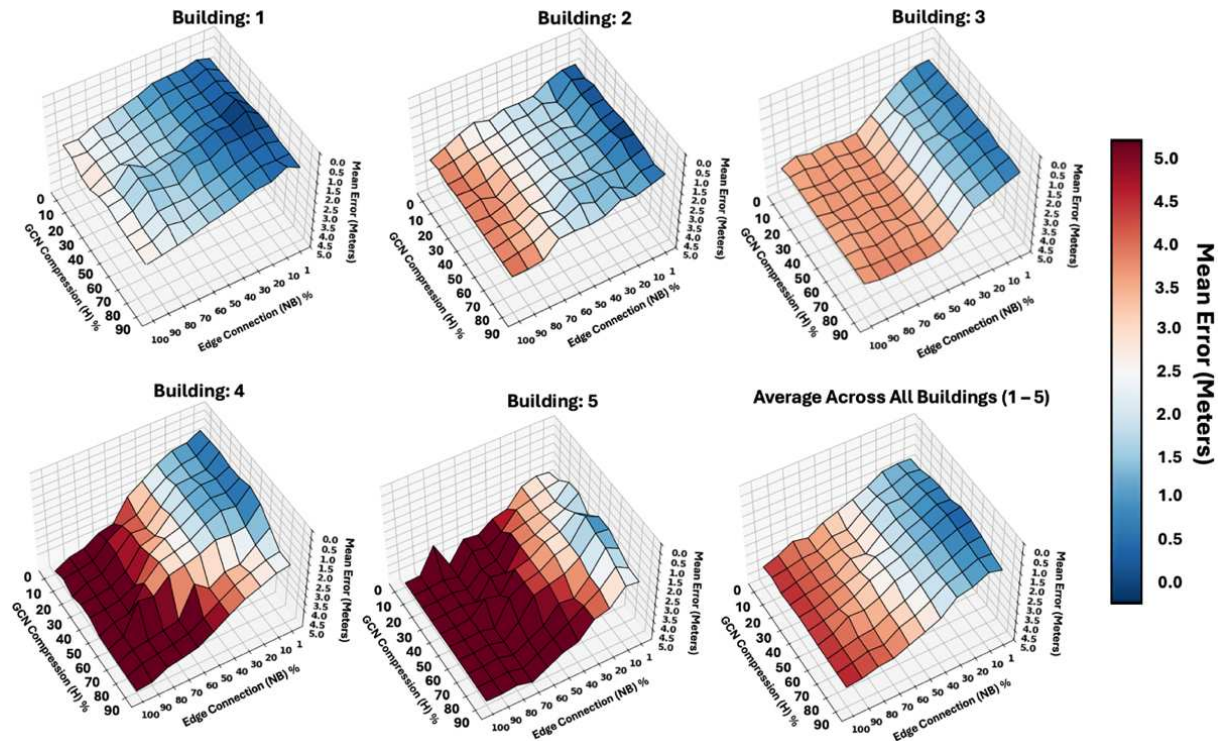


Figure 113: Sensitivity of GATE to GCN compression size (H%) and edge connectivity (NB%) for each building floorplan.

The number of edges per node (NB) has a more pronounced impact on the GATE framework's behavior. Across all buildings, we observe a clear trend: increasing NB leads to progressively higher localization errors. This degradation is due to the GNN blind spot problem, where excessive

connectivity introduces representational noise, especially in dense graphs. While AHV provides per-feature attention weighting to mitigate this effect, it cannot entirely resolve the dilution of spatial specificity when irrelevant nodes are aggregated. Interestingly, GATE shows strong resilience to this blind spot problem in Buildings 1 and 2, where despite very high AP densities, errors remain below 2 meters up to $NB = 60\%$. This is a direct result of the MDHV structure, which leverages both AHV and MSG to model inter-node relevance and feature-wise contrasts effectively. Even in Building 3, GATE maintains stable performance up to $NB = 40\%$, which is notably higher than what traditional GNNs can tolerate in similar settings (see experimental analysis in Section 12.6.6). In contrast, the blind spot problem becomes more apparent in Buildings 4 and 5, with performance deterioration observed at much lower NB values—around 40% and 20%, respectively. These environments lack the high-dimensional input space needed for AHV and MSG to construct more representative data embeddings, making them more sensitive to edge noise. Despite this, GATE still outperforms conventional baselines by a significant margin, as will be discussed in Section 12.6.6. The sixth, aggregated plot provides a generalized view: maintaining NB at or below 10% and setting H to approximately 50% yields consistently low localization error across all environments. This configuration is the one we consider for subsequent analysis as it allows GATE to significantly mitigate the GNN blind spot problem in indoor localization.

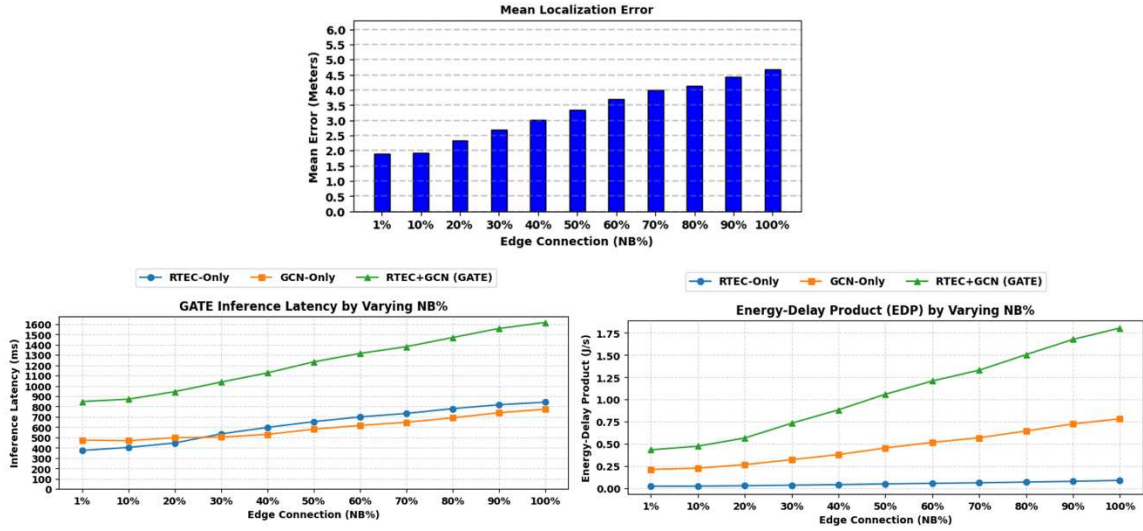


Figure 114: Impact of edge connectivity (NB%) on GATE’s localization accuracy, inference latency, and energy-delay product (EDP) across all mobile devices and buildings.

To evaluate the real-world deploy ability of GATE, we analyze its performance across all mobile devices and buildings presented in Figure 112, focusing on how NB% impacts both localization accuracy and on-device efficiency. We consider two critical deployment metrics: inference latency—the time required to generate location predictions—and energy-delay product (EDP)—a measure of the energy consumed to perform inference. These metrics are evaluated under three configurations: 1) RTEC-Only, where we capture the time and energy required to compute attention-based edge connections and construct the MDHV; 2) GCN-Only, which measures the cost of performing inference on the MDHV; and 3) RTEC+GCN, which reflects the total end-to-end latency and energy consumption of the full GATE pipeline. Figure 114 presents the results: the top bar plot shows the mean localization error (in meters) for each NB%, averaged over all mobile devices and buildings. The bottom-left and bottom-right line plots show the corresponding average inference latency (in milliseconds) and EDP (in joules per second),

respectively. In all plots, the X-axis represents NB%, while the Y-axis denotes the respective performance metrics.

In the RTEC-Only configuration, inference latency remains low and stable for NB values between 1% and 20%, but increases progressively beyond this point, reaching a maximum of 842 ms at NB = 100%. This trend is expected, as higher NB values lead to an increased number of attention score calculations and message-passing operations required to construct the MDHV. Despite this rise in latency, the EDP remains relatively flat across all NB settings, ranging from 0.06 J/s at NB = 1% to 0.11 J/s at NB = 100%. This suggests that although execution time grows with higher edge connectivity, energy consumption does not rise proportionally. The efficiency of RTEC stems from its use of lightweight scalar dot-product attention operations, which are less compute-intensive than the dense matrix operations typical of convolutional layers. As a result, RTEC remains highly energy-efficient and well-suited for execution on mobile devices.

In the GCN-Only configuration—which is executed after the MDHV are constructed—we observe that inference latency remains nearly constant for NB values between 1% and 40%, with only a slight increase beyond that. GCN inference is consistently faster than RTEC, as the GCN does not need to compute attention scores individually for every node during inference; instead, it operates on the pre-computed MDHV (from RTEC), which is highly optimized on modern mobile device hardware. However, the EDP increases noticeably at higher NB values, especially beyond 30%. This increase is driven by the growing size of the MDHV in denser graphs. Larger input dimensions result in more intensive feature-wise computations within each convolutional layer, leading to higher energy consumption despite relatively stable runtime. When evaluating the full RTEC+GCN pipeline, we find that total inference latency increases from 848 ms at NB = 1% to 1616 ms at NB = 100%, while EDP rises from 0.5 J/s to 1.11 J/s over the same range. Despite this

growth, both metrics remain within acceptable thresholds for real-time inference on modern mobile devices, demonstrating GATE’s practicality for real-time deployments. In summary, our analysis identifies NB values between 1% and 10% as the optimal operating range for real-world use. Within this range, GATE consistently delivers localization errors below 2 meters, end-to-end inference latency under 1 second, and EDP below 0.6 J/s across all devices and buildings. These findings affirm that GATE is not only accurate and robust but also energy-efficient for deployment on resource-constrained mobile devices.

12.6.2. SENSITIVITY ANALYSIS: EFFECTS OF FINGERPRINT SAMPLES PER RP ON HETEROGENEITY RESILIENCE

In this section, we evaluate the impact of varying the number of fingerprint samples collected per RP, and how this influences the GATE framework’s ability to generalize across heterogeneous devices. Specifically, we vary the number of training samples per RP from 1 to 5, in increments of 1, while keeping the test configuration fixed at one fingerprint sample per RP. Figure 115 presents the results across all seven smartphones, with separate plots for each building and an aggregated plot that captures the overall trend. The X-axis denotes the seven devices, and the Y-axis denotes the mean localization error in meters. As illustrated in Figure 115, increasing the number of training samples per RP consistently reduces the mean localization error and substantially lowers device variance—quantified as the difference between the highest and lowest localization error among devices. Our analysis reveals those buildings with higher AP densities—specifically Buildings 1, 2, and 3—exhibit lower worst-case localization errors. For instance, with five samples per RP, these buildings achieve mean localization errors of 1.36, 1.43, and 1.84 meters, respectively across all devices. In contrast, Buildings 4 and 5, which have lower AP

densities, observe higher errors of 2.39 and 2.41 meters, respectively across all devices. When examining the aggregated results across all buildings, training with only one sample per RP yields the highest device variance: the mean localization error ranges from 3.84 meters (worst-performing device) to 2.35 meters (best-performing device), reflecting a variance of 1.49 meters. In comparison, training with 2, 3, 4, and 5 samples per RP results in reduced variances of 1.15, 1.07, 0.5, and 0.25 meters, respectively. Notably, training with five samples per RP achieves the lowest device variance, representing a $1.83\times$ improvement in heterogeneity resilience compared to the single-sample case.

This enhanced resilience is attributed to the higher RSS fingerprint variability obtained from multiple samples per RP. When more samples are used during training, the resulting RSS values introduce subtle but meaningful variability for each node, which subtly improves the variability of the MSG vector. This broader RSS distribution allows the AHV to capture fine-grained feature-wise variability across node edges, leading to a more diverse and robust MDHV representation. Consequently, the GATE framework becomes better equipped to generalize to devices with differing signal characteristics. Empirically, this results in a mean localization error of 1.98 meters across all test devices and buildings when using five samples per RP. Based on this analysis, we fix the number of training samples per RP to five for all subsequent experiments.

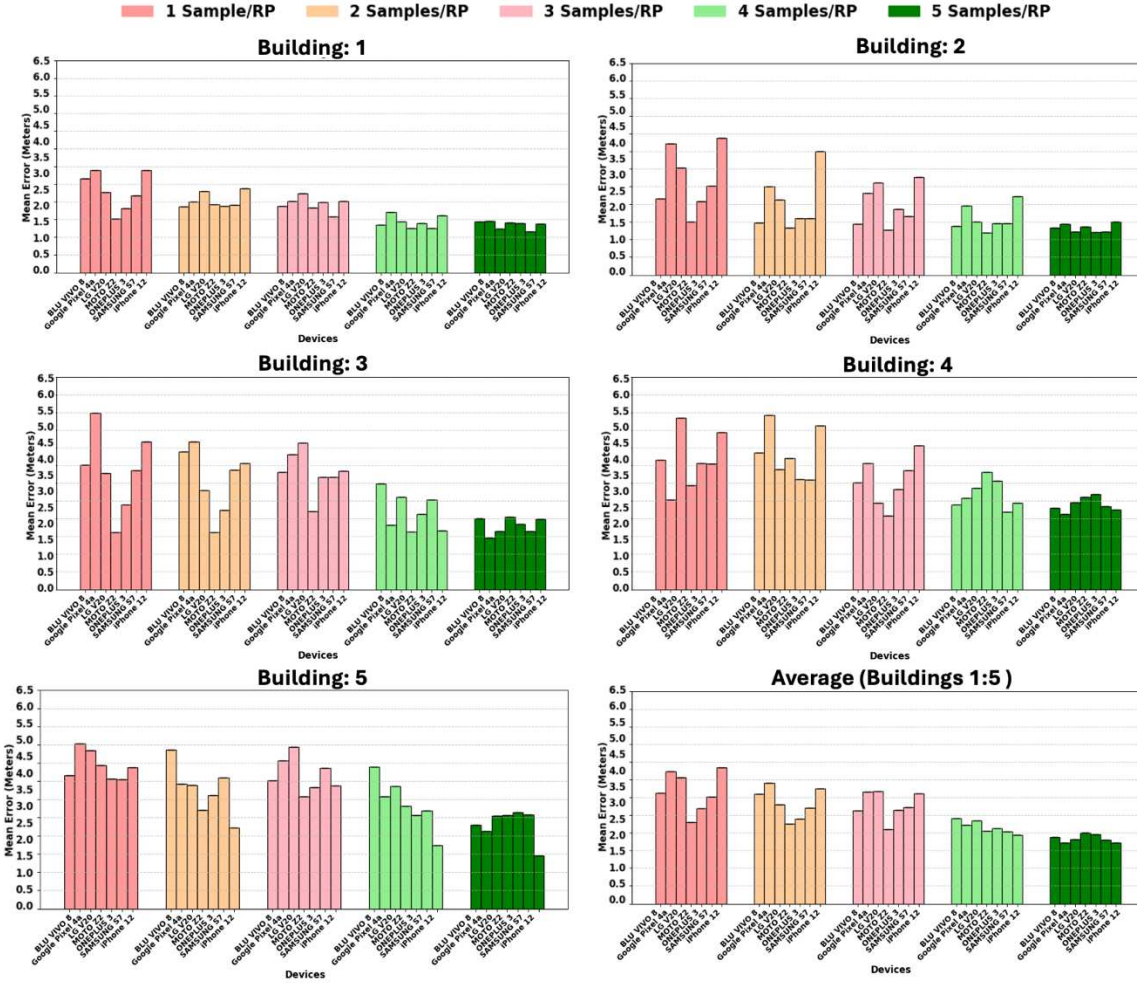


Figure 115: Sensitivity of GATE to the number of fingerprint samples per RP during training.

12.6.3. COMPONENT-WISE ANALYSIS OF GATE: IMPACT ON WORST-CASE ERRORS AND HETEROGENEITY RESILIENCE

Next, we assessed the contributions of each major component in the GATE framework by comparing four architectural variants: 1) GATE-No-MSG which removes the MSG vector and constructs the MDHV using only the fingerprint vector and AHV during both training and RTEC-based inference, 2) GATE-No-AHV which excludes the AHV while retaining the fingerprint vector and MSG vector during both training and RTEC-based inference, 3) GATE-No-MDHV

which removes both MSG and AHV, relying solely on the fingerprint vector as input and disabling RTEC during inference, and 4) GATE-Full which uses all three components—fingerprint vector, MSG, and AHV—within the MDHV during both training and RTEC-based inference. Figure 116 presents the results across all seven devices, averaged over the five building floorplans. The X-axis denotes the testing devices, and the Y-axis denotes the mean localization error in meters. The result for each device is shown as a box plot, where the red line represents the mean error across all RPs and building floorplans, the upper whisker indicates the worst-case error across all RPs for that device, and the lower whisker shows the best-case error across all RPs for that device.

From Figure 116, we observe that GATE-No-MDHV performs the worst among all variants, exhibiting the highest worst-case error and the highest susceptibility towards heterogeneity. The Samsung S7 records the largest outlier at 8.3 meters, while the device variance—defined as the difference in mean localization error between the best and worst performing devices—peaks at 2.25 meters, ranging from 4.75 meters on the BLU Vivo 8 to 2.5 meters on the Moto Z2. This highlights the importance of the MDHV in managing environmental noise and device-induced signal variability. GATE-No-AHV results in the second-highest worst-case error of 7.15 meters (Google Pixel 4a) but shows improved resilience with a lower device variance of 1.1 meters, suggesting that the presence of the MSG vector helps capture inter-node context and stabilizes performance across devices. However, the absence of AHV limits the model’s ability to account for feature-wise irregularities, leaving it vulnerable to localized noise. In contrast, GATE-No-MSG reduces the worst-case error slightly to 5.8 meters but suffers from high device variance (2.17 meters), ranging from 3.27 meters (BLU Vivo 8) to 1.1 meters (Moto Z2). This indicates that while AHV helps suppress extreme errors, omitting MSG weakens the model’s generalization across devices. GATE-Full achieves the best results overall, with the lowest worst-case error of 3.2 meters

(OnePlus 3T) and minimal device variance of 0.25 meters. These findings confirm that the combination of MSG and AHV within MDHV enables our framework to learn both edge-level influence and feature-level contributions, preserving the non-Euclidean structure and achieving robustness towards device heterogeneity and environmental noise.

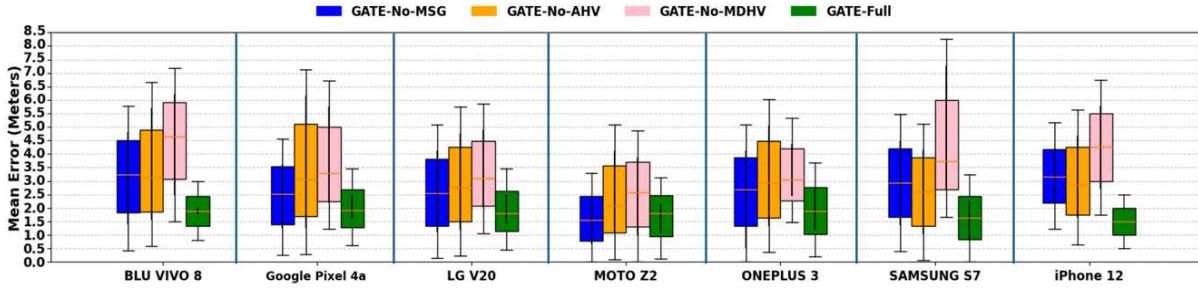


Figure 116: Component-wise analysis of different GATE variants.

12.6.4. IMPACT OF VARYING AP DENSITIES ON GATE VARIANTS: PERFORMANCE ACROSS TRUNCATED FINGERPRINT VECTOR SIZE

We also investigated how the performance of the four GATE variants—GATE-No-MDHV, GATE-No-AHV, GATE-No-MSG, and GATE-Full—varies with the dimension of the fingerprint vectors per RP. We simulated environments with limited AP visibility by progressively truncating each fingerprint vector, randomly dropping a percentage of RSS values per RP while keeping the graph connectivity intact. This experiment assesses the robustness of each GATE variant in situations where fewer APs are visible. In Figure 117, the X-axis denotes fingerprint vector truncation percentage varied and the Y-axis denotes the mean localization error (in meters), averaged across all RPs, buildings, and devices.

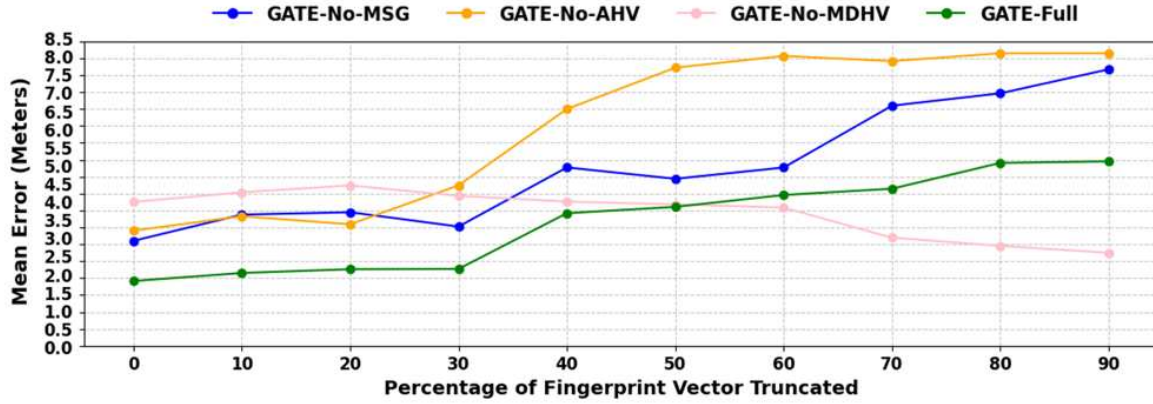


Figure 117: Impact of varying AP densities (fingerprint vector truncation) on localization accuracy across GATE variants.

The GATE-No-AHV variant shows the sharpest increase in mean localization error as the fingerprint vector is truncated. Performance remains stable up to 20% truncation, but errors rise significantly between 30% and 60%, plateauing at higher truncation levels. This behavior highlights the importance of AHV in capturing per-feature contributions during message passing. Without AHV, the MSG vector aggregates larger noise from neighboring nodes, especially when fewer RSS features are available, reducing discriminative capacity. The GATE-No-MSG variant performs slightly better but still shows marked error increases beyond 30% and again around 60%. While AHV offers some local feature sensitivity, its effectiveness declines with reduced input dimensionality as the attention scores rely on feature variation to assign meaningful weights, when dimensions are heavily truncated, this variation diminishes, making the scores less discriminative. The GATE-Full variant yields lower errors across all truncation levels compared to GATE-No-AHV and GATE-No-MSG, with only mild increases around the 40% and 80% marks, and maintains a bounded error of 5.5 meters at 90% truncation. Interestingly, the GATE-No-MDHV variant shows a counterintuitive trend: localization error decreases as truncation increases beyond 50%. This behavior likely arises because lower-dimensional inputs simplify the learning task for

models without contextual aggregation, especially in simpler layouts or low-variability settings. However, such configurations lack robustness and fail under heterogeneous or complex environments with higher AP densities. These findings suggest that in scenarios with small fingerprint vectors (i.e., low AP density), simpler GNNs/GCNs may suffice, as message passing introduces noise that outweighs its benefits. However, as AP density increases and the environment becomes more complex, message passing alone becomes unreliable due to the GNN blind spot problem. In these cases, the MDHV becomes critical, preserving both contextual and per-feature signal variation, thereby effectively mitigating the GNN blind spot problem in high-dimensional settings.

12.6.5. COMPARISON OF GATE VARIANTS WITH STATE-OF-THE-ART INDOOR LOCALIZATION FRAMEWORKS

Lastly, we compare the performance of GATE and its variants against state-of-the-art GNN-based localization frameworks from prior work. Figure 118 summarizes the results, where the X-axis denotes the evaluated frameworks and the Y-axis indicates mean localization error in meters, averaged across five building floorplans and seven mobile devices. In each box plot, the upper whisker denotes the worst-case error, the lower whisker denotes the best-case error, and the orange line denotes the mean error. As can be observed in Figure 118, GATE-Full achieves the lowest mean localization error of 1.98 meters and the lowest worst-case error of 3.5 meters across all building floorplans and devices, outperforming all current state-of-the-art frameworks. The second-best performance comes from GATE-No-MSG, which lacks the message passing (MSG) vector. In comparison, GATE-Full yields $1.38\times$ lower mean error and $1.61\times$ lower worst-case error, demonstrating the importance of MSG for device heterogeneity resilience. The GCLoc [240]

and GraphLoc [244] frameworks from prior work rank third and fourth due to their use of Graph Attention (GAT) based architectures. However, their lack of AHV makes them unable to capture the non-Euclidean feature structure of fingerprint vectors, especially in high AP density environments. GATE-Full surpasses GCLoc and GraphLoc with $1.6\times$ to $1.8\times$ lower mean error and $1.8\times$ to $1.9\times$ lower worst-case error respectively. GATE-No-AHV performs similarly to GCLoc in terms of worst-case error but has a slightly higher mean error, with GATE-Full achieving $1.73\times$ lower mean error in comparison. Traditional models such as GNN-KNN [238], GCN-KNN [239], GNN-ED [236], GCN-ED [237], STONE [32], STELLAR [134], SANGRIA [82], DNN [42], CNN [6], and KNN [229] show significantly higher mean and worst-case errors. These frameworks are limited by fixed graph structures and Euclidean assumptions, making them vulnerable to signal variability and the GNN blind spot effect. GATE-Full achieves $2.26\times$, $2.24\times$, $2.48\times$, $3.31\times$, $3.03\times$, $3.4\times$, $3.9\times$, $4.12\times$, $4.42\times$, and $4.72\times$ lower mean errors and $2.62\times$, $4.00\times$, $3.22\times$, $4.05\times$, $2.56\times$, $2.61\times$, $3.22\times$, $4.28\times$, $4.28\times$, and $4.57\times$ lower worst-case errors respectively. These results indicate that integrating both the MSG vector and AHV within the MDHV not only preserves the non-Euclidean structure of RSS fingerprints but also mitigates noise propagation during message passing—particularly in high-dimensional fingerprint environments—effectively mitigating the GNN blind spot effect and enhancing overall localization performance.

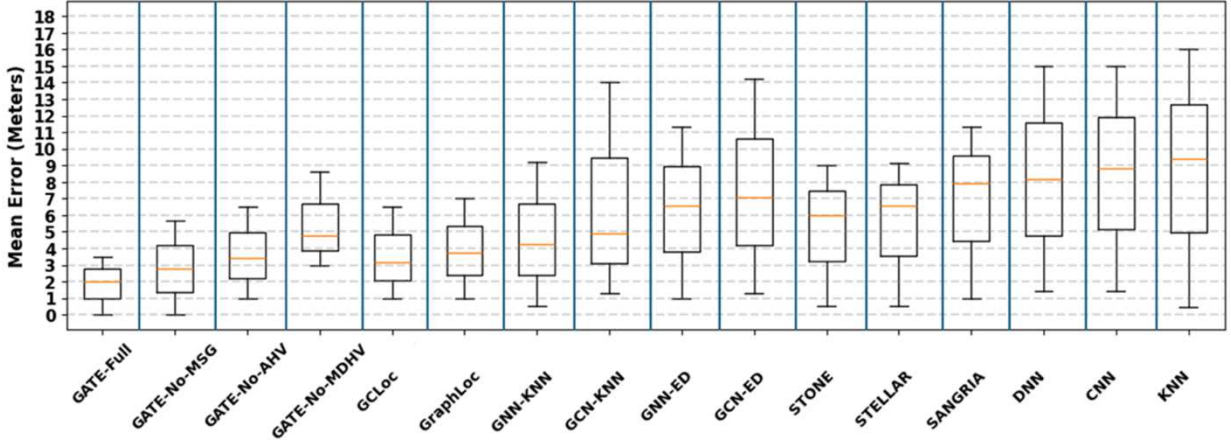


Figure 118: Evaluation of the GATE variants compared to state-of-the-art indoor localization frameworks.

Next, we evaluate the real-world deployability of GATE and the baseline frameworks on the mobile devices detailed in Figure 112. We analyze several key hardware metrics presented in Table 1—averaged across all seven devices and five buildings—to understand each model’s suitability for mobile deployment. These metrics include: 1) Inference Latency: The time required for a single location prediction, measured in milliseconds (ms). Lower latency is essential for real-time applications such as indoor localization. 2) Model Size: This is the total memory footprint of the trained model, reported in kilobytes (KB). Models with smaller memory footprints are easier to deploy on mobile devices with limited storage and memory constraints. 3) Energy-Delay Product (EDP): This metric captures energy efficiency by measuring the energy consumed per prediction, measured in Joules per second (J/s). Lower EDP indicates the model is energy-efficient on battery-powered devices. 4) FLOP ($\times 10^3$): This measures the total number of floating-point operations required for a single inference, reflecting the model’s computational complexity. A higher FLOP value typically corresponds to models being more computationally demanding.

Table 20: Comparison of hardware performance metrics (averaged across all seven devices and five buildings) for GATE and baselines.

Model	Mean Error (Meters)	Inference Latency (ms)	Model Size (KB)	EDP (J/s)	FLOP ($\times 10^3$)
GATE-No-MSG	2.75	836	599.3	0.521	576.8
GATE-No-AHV	3.45	799	590.4	0.493	319.6
GATE-No-MDHV	4.75	734	509.3	0.444	271.5
GATE-Full	1.98	871	604.0	0.543	714.2
GCLoc	3.19	904	1404.8	0.53	786.4
GRAPHLOC	3.75	1034	2234.1	0.75	982.3
GNN-KNN	4.25	934	934.8	0.61	644.4
GCN-KNN	4.9	1077	1034.0	0.65	796.9
GNN-ED	6.6	698	498.7	0.50	349.0
GCN-ED	7.1	754	511.3	0.52	414.7
SANGRIA	7.89	437	1774.3	0.47	262.2
STONE	6.0	1301	5667.1	1.01	1301.0
STELLAR	6.6	1554	8024.0	1.20	1864.8
DNN	8.2	855	204.2	0.40	282.1
CNN	8.8	1231	636.3	0.73	726.2
KNN	9.4	1531	1224.6	0.93	688.9

Starting with GATE-Full, this architecture achieves a balance between accuracy and hardware efficiency. As shown in Table 20, it achieves the lowest mean localization error of 1.98 meters—substantially outperforming all baselines—while maintaining a moderate latency of 871 ms and an EDP of 0.543 J/s. Its compact model size of 604.0 KB ensures easy deployment on mobile devices with limited storage. The relatively higher FLOP value (714.2×10^3) reflects the added computational steps required by the integrated MSG vector and AHV modules, which captures the influence of edges and maintain the non-Euclidean RSS structures. This moderate increase in computational workload is a deliberate trade-off that enhances robustness against device heterogeneity and environmental noise. Comparatively, GATE-No-MSG, GATE-No-AHV, and GATE-No-MDHV achieve slightly lower latencies (836 ms, 799 ms, and 734 ms) and EDPs (0.521

J/s, 0.493 J/s, and 0.44 J/s) due to the absence of the respective modules; however, they suffer from increased mean errors (2.75, 3.45, and 4.75 meters), highlighting the importance of retaining both components for consistent performance.

Examining the baselines reveals key insights: graph attention-based (GAT) frameworks like GCLoc [240] and GraphLoc [244] exhibit significantly larger model sizes (1404.8 KB and 2234.1 KB, respectively) because they lack the RTEC module for selective node filtering, forcing them to store the entire graph during inference. This results in GATE-Full being $2.32\times$ to $3.69\times$ more compact while also achieving lower latency and EDP, as shown in Table 20. GraphLoc’s multi-headed attention further inflates its FLOP value (982.3×10^3), EDP (0.75 J/s), and latency (1034 ms), while GCLoc, despite a slightly lower EDP (0.53 J/s), suffers from higher mean errors due to its inability to efficiently curate message-passing paths without RTEC. Other graph-based baselines like GNN-KNN [238], GCN-KNN [239], GNN-ED [236], and GCN-ED [237] require lower FLOP ($349.0\text{--}796.9\times 10^3$) and achieve lower EDP (0.50–0.65 J/s) but consistently higher errors (4.25–7.1 meters) due to the lack of the AHV module, which is essential for modeling feature-wise node influence for device heterogeneity. Traditional baselines like DNN [42], CNN [6], and KNN [229] show lower model sizes (204.2–636.3 KB) and FLOP values ($282.1\times 10^3\text{--}726.2\times 10^3$) but still suffer from higher errors (8.2–9.4 meters), as they cannot capture the non-Euclidean structure of RSS fingerprints effectively. STONE [32], STELLAR [134], and SANGRIA [82] stand out with extremely large model sizes (1774.3–8024 KB), very high latencies (437–1554 ms), and EDPs exceeding 1.0 J/s, rendering them impractical for mobile deployments. In summary, GATE-Full consistently achieves sub-2-meter mean errors, inference latency of less than 1 second, and efficient EDP of less than 0.6 J/s, across all buildings and mobile devices, demonstrating robust and practical hardware deployment on mobile devices.

12.7. CONCLUSIONS

In this chapter, we introduced GATE, a novel graph-based indoor localization framework designed to: 1) mitigate both environmental noise and device heterogeneity by preserving the inherently non-Euclidean structure of fingerprint vectors, 2) maintain indoor topology within the graph structure to ensure spatial context is retained, and 3) overcome the GNN blind spot problem by enabling robust message passing even in high-density Wi-Fi AP environments. GATE integrates attention-based message passing and the Attention Hyperspace Vector (AHV) into its Multi-Dimensional Hyperspace Vector (MDHV) representation, while employing Real-Time Edge Construction (RTEC) during inference to dynamically form meaningful graph structures that closely represent the characteristics of the indoor space. This enables GATE to maintain spatial relationships between indoor locations, encode feature-level signal variation, and preserve the non-Euclidean structure of fingerprint vectors. Extensive experiments across diverse indoor environments and heterogeneous devices demonstrate that GATE outperforms state-of-the-art indoor localization frameworks, achieving $1.6\times$ to $4.72\times$ lower mean localization errors and $1.85\times$ to $4.57\times$ lower worst-case errors.

CHAPTER 13

TOWARDS EXPLAINABLE INDOOR LOCALIZATION: INTERPRETING NEURAL NETWORK LEARNING ON WI-FI FINGERPRINTS USING LOGIC GATES

Indoor localization using deep learning (DL) has demonstrated strong accuracy in mapping Wi-Fi RSS fingerprints to physical locations; however, most existing DL frameworks function as black-box models, offering limited insight into how predictions are made or how models respond to real-world noise over time. This lack of interpretability hampers our ability to understand the impact of temporal variations—caused by environmental dynamics—and to adapt models for long-term reliability. To address this, we introduce LogNet, a novel logic gate-based framework designed to interpret and enhance DL-based indoor localization. LogNet enables transparent reasoning by identifying which access points (APs) are most influential for each reference point (RP) and reveals how environmental noise disrupts DL-driven localization decisions. This interpretability allows us to trace and diagnose model failures and adapt DL systems for more stable long-term deployments. Evaluations across multiple real-world building floorplans and over two years of temporal variation show that LogNet not only interprets the internal behavior of DL models but also improves performance—achieving up to $1.1\times$ to $2.8\times$ lower localization error, $3.4\times$ to $43.3\times$ smaller model size, and $1.5\times$ to $3.6\times$ lower latency compared to prior DL-based models.

13.1. INTRODUCTION

Indoor localization has become a cornerstone of modern context-aware technologies, enabling applications in robotics, augmented and virtual reality (AR/VR), asset tracking, and emergency response. One of the earliest indoor localization system, "*The Active Badge Location System*"

introduced in 1992 [224], relied on infrared (IR) pulses emitted by wearable badges and captured by stationary IR receivers [224]. However, its reliance on line-of-sight (LoS) for IR communications rendered it ineffective in large-scale environments. As indoor environments have grown more dynamic and complex, researchers have increasingly turned to artificial intelligence (AI) to enhance localization robustness and scalability [245]. This paradigm shift has driven widespread adoption across industries, with companies such as Apple, Google, Microsoft, HPE, Quuppa, and Zebra investing heavily in AI-powered indoor localization systems [34]. Reflecting this momentum, the global indoor localization market was valued at \$11.9 billion in 2024, with strong forecasts for continued growth driven by the demand for high-precision and intelligent indoor localization frameworks [246].

Among AI techniques, neural network based Deep Learning (DL) models have shown promise in capturing spatial patterns within Radio Frequency (RF) data for indoor localization [64]. While indoor localization systems may use various RF technologies—such as Bluetooth, Ultra-Wideband (UWB), or cellular signals—Wi-Fi remains the most widely adopted due to its pervasive presence and seamless integration with mobile devices. Wi-Fi-based localization relies on constructing high-dimensional "fingerprints" from Received Signal Strength (RSS) values measured across multiple Access Points (APs) per Reference Point (RP) location [45]. DL models have been favored for their ability to learn non-linear decision boundaries across high-dimensional RSS fingerprints. However, they typically function as opaque black-box systems, offering limited interpretability into how RSS from specific APs can influence localization predictions [247].

This black-box assumption becomes problematic in dynamic indoor environments, where noise introduced by factors such as human movement, signal shadowing, and multipath signal propagation can degrade model accuracy [134]. While some of this noise may have short-term

effects, long-term environmental changes such as permanent furniture rearrangement and aging/replacement of Wi-Fi transceivers lead to what is known as temporal variations that distort the RSS fingerprint distributions over time [245], [134]. In DL, these variations can cause learned decision boundaries between RPs to break down, leading to higher localization errors and unreliable long-term performance [248]. More critically, the lack of interpretability in DL models leaves us unable to identify *Which APs are most influential for a given RP?* and *How does the model form decision boundaries between RPs?* Without answers to these questions, diagnosing failures and improving model robustness becomes exceptionally challenging.

13.2. CONTRIBUTIONS

To address these important issues, we introduce LogNet, a novel logic gate-based network that not only interprets the internal representations of DL models but also enhances their robustness to temporal variations. LogNet leverages logic gate operations to reveal the most influential APs per RP and exposes how environmental changes disrupt classification. These insights enable us to enhance DL models and extend their usability over time. Our work aims to answer three key research questions: **RQ1** – *What underlying factors cause DL-based indoor localization models to deteriorate under temporal variation?* **RQ2** – *How can identifying the most influential APs for each RP help in enhancing model robustness against temporal variations?* **RQ3** – *In what ways can logic gate networks be employed to both interpret DL decision processes and improve their resilience to temporal variations?* Our novel contributions are as follows:

- We introduce LogNet, a framework that brings interpretability to DL-based indoor localization.
- We demonstrate how LogNet can be used to identify failures in DL models due to temporal variation and improve their resilience.

- We perform extensive evaluations on real-world datasets collected over two years across buildings, showing that LogNet outperforms existing DL methods in both interpretability and long-term localization performance.

13.3. RELATED WORKS

Indoor localization techniques have progressed rapidly over the past decade, evolving from traditional geometric methods to DL-based approaches, driven by the need for greater accuracy and robustness in real-world environments. Classical systems such as TDoA-Loc [249], AngLoc [250], RSSI-Trilat [251], and BGI-Trilateration [252] leveraged geometric principles—like time-difference-of-arrival (TDoA) and angle-of-arrival (AoA)—to estimate a user’s position relative to known RPs. These models offered spatially interpretable decision boundaries, making them attractive for explainable deployments. However, their practical utility was constrained by the need for clear LoS, susceptibility to multipath propagation, and signal shadowing—common challenges in indoor environments.

To address these limitations, fingerprinting-based methods emerged, including RADAR [253]. These approaches construct a database of RSS fingerprints collected across RPs and use similarity metrics such as k-nearest neighbors (k-NN) or Euclidean distance (ED) to estimate positions. Compared to geometric methods, fingerprinting techniques demonstrated better adaptability in non-LoS conditions. However, fingerprinting techniques lack the interpretability of geometric models. Since prediction is based on similarity matching, it is difficult to isolate the contribution of an individual AP to localization decisions [248]. Furthermore, these methods assume noise affects all APs uniformly—i.e., a Euclidean view of the fingerprint vector. In contrast, real-world

noise is often non-uniform across APs, leading to a more complex, non-Euclidean distribution (further discussed in Section 13.4).

To overcome these shortcomings, recent advancements have turned to DL algorithms, particularly deep neural networks (DNNs) [88], [258], [259], which are adept at learning complex nonlinear mappings from high-dimensional RSS fingerprints to RP locations. Several models—such as DNNLOC [78], ISDL [254], and HWF [255]—have demonstrated improved accuracy over similarity-based fingerprinting methods. These DNNs typically adopt one of three core architectural patterns [256]: 1) DNN-Dense, where all hidden layers maintain uniform dimensionality; 2) DNN-UpSample, which progressively increases dimensionality in deeper layers; and 3) DNN-DownSample, where hidden layers compress the feature space successively. Despite these advancements, all DNN-based architectures share a key limitation: performance deterioration over long-term temporal variations. These variations which occur over time (weeks to months after initial deployment) disrupt the learned latent space representations and decision boundaries between RP classes. Compounding this issue is the black-box nature of DNNs, which hinders interpretability and makes it difficult to identify which APs are most influential for any given RP [257]. Moreover, DNNs often implicitly process fingerprint vectors under the assumption of uniform (Euclidean) noise, further exacerbating their failure under real-world, non-Euclidean conditions. Addressing this challenge requires a deeper understanding of how DNNs encode RSS data in their latent space (decision boundary space between classes) and how these encodings evolve with time. This insight is essential not only for diagnosing model failures but also for improving resilience against temporal variation.

13.4. INTERPRETING DL FOR INDOOR LOCALIZATION

In this section, we first analyze the structure of noise distribution in Wi-Fi RSS fingerprint vectors caused by temporal variations and demonstrate how conventional assumptions of uniformity fail in real-world settings. We then introduce logic gates as interpretable computational primitives and explore how their behavior can be harnessed to understand and model the internal learning processes of DNNs for indoor localization.

13.4.1. STRUCTURAL ANALYSIS OF WI-FI RSS FINGERPRINTS

As a first step, we analyze the structural characteristics of Wi-Fi RSS fingerprints (F) and the temporal noise patterns that arise over time due to environmental dynamics. Traditionally, F is assumed to follow a Euclidean (ED) structure, where noise is distributed uniformly across all APs. However, real-world deployments often exhibit non-uniform noise patterns, leading to a non-Euclidean (non-ED) structure. As illustrated in Figure 119 and formalized in Equation 83, in an ED structure, a constant noise term (δ) is assumed and applied across all APs. In contrast, Equation 84 describes the non-ED structure, where each AP is subject to its own unique noise term, capturing the uneven impact of temporal variations across APs.

$$F_{ED} = [(\delta, RSS_{AP:0}), (\delta, RSS_{AP:1}), \dots, (\delta, RSS_{AP:N})] \quad (83)$$

$$F_{Non-ED} = [(\delta_0, RSS_{AP:0}), (\delta_1, RSS_{AP:1}), \dots, (\delta_N, RSS_{AP:N})] \quad (84)$$

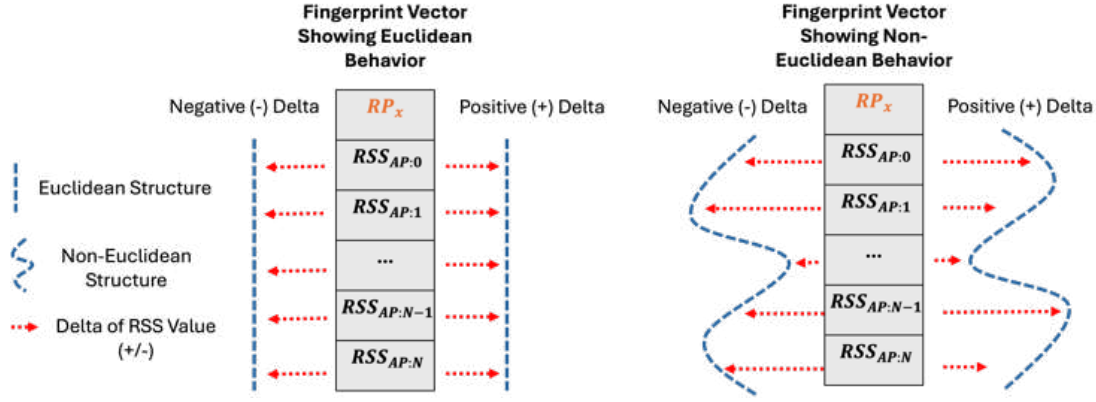


Figure 119: Euclidean vs. Non-Euclidean fingerprint structures.

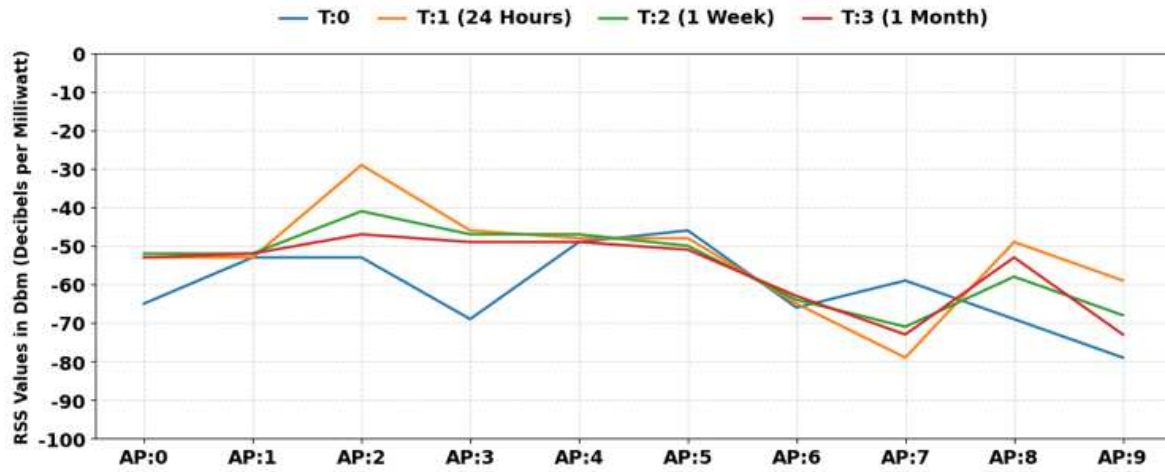


Figure 120: Temporal variation of RSS values across the first 10 APs.

To empirically verify this non-ED behavior, we collected RSS fingerprints from the same smartphone device and RP over four time points: T:0 (initial), T:1 (after 24 hours), T:2 (after 1 week), and T:3 (after 1 month). Figure 120 shows the temporal fluctuations in RSS across the first 10 APs within F. While APs like AP:1, AP:4, AP:5, and AP:6 remain relatively stable, others display significant variation, confirming the presence of non-ED noise. This structural insight challenges the ED assumptions inherent in most DNN-based localization models.

Table 21: Truth Table of logic gates

AND Gate			OR Gate			NAND Gate		
X	Y	Z	X	Y	Z	X	Y	Z
0	0	0	0	0	0	0	0	1
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

NOR Gate			XOR Gate			XNOR Gate		
X	Y	Z	X	Y	Z	X	Y	Z
0	0	1	0	0	0	0	0	1
0	1	0	0	1	1	0	1	0
1	0	0	1	0	1	1	0	0
1	1	0	1	1	0	1	1	1

13.4.2. LOGIC GATES FOR INDOOR LOCALIZATION

To enable interpretable reasoning in DL-based indoor localization, we introduce logic gates as foundational components for decision-making. Each gate receives two inputs derived from the RSS fingerprint vector—representing whether two APs are active (1) or inactive (0) based on a defined threshold (discussed in Section 13.5). These gates evaluate binary relationships between pairs of APs and form the basis of simple and transparent decision rules that can be visualized and interpreted in real time. The truth Table for each logic gate is shown in Table 21. The behavior of various types of gates that we considered are summarized below:

- **AND Gate:** Outputs 1 only when both adjacent APs are active, distinguishing regions in the fingerprint, where both APs exhibit strong RSS patterns (co-occurrence) from those where they do not.

$$Z = X.Y \quad (85)$$

- **OR Gate:** Outputs 1 if at least one of the adjacent APs is active, distinguishing regions in the fingerprint, where there is some AP visibility and separates them from dead zone patterns (both APs showing weak RSS).

$$Z = X + Y \quad (86)$$

- **NAND Gate:** Outputs 1 unless both adjacent APs are active, distinguishing regions in the fingerprint, where at least one AP is weak and separate zones with co-occurrence patterns.

$$Z = 1 - (X.Y) \quad (87)$$

- **NOR Gate:** Outputs 1 when both adjacent APs are inactive, distinguishing regions in the fingerprint showing dead zones patterns from those where they do not.

$$Z = 1 - (X + Y) \quad (88)$$

- **XOR Gate:** Outputs 1 when exactly one of the two APs is active, distinguishing regions in the fingerprint, where only one AP is strong and separates regions showing co-activation and dead zone patterns.

$$Z = X \cdot (1 - Y) + (1 - X) \cdot Y \quad (89)$$

- **XNOR Gate:** Outputs 1 when both APs are in the same state—either both active or both inactive, distinguishing regions in the fingerprint, where the APs either show co-activation or dead zone patterns.

$$Z = X \cdot Y + (1 - X) \cdot (1 - Y) \quad (90)$$

By mapping binarized RSS relationships through these gates, we uncover interpretable logic patterns that contribute to a location decision. These gates form the building blocks of the proposed LogNet architecture, allowing us to reason about how RSS patterns are internally processed and how decision boundaries emerge, as discussed next.

13.5. THE LOGNET FRAMEWORK

The LogNet framework introduces an interpretable alternative to conventional DNN architectures for indoor localization by constructing a multi-layer network composed entirely of binary logic gates. As discussed in Section 13.4, each gate operates on a pair of binarized RSS values—designated as either active (1) or inactive (0)—and performs a predefined logical operation (e.g., AND, OR, NAND). This architecture enables LogNet to offer rule-based reasoning

while learning spatial relationships among APs in a transparent and diagnosable (traceable) manner.

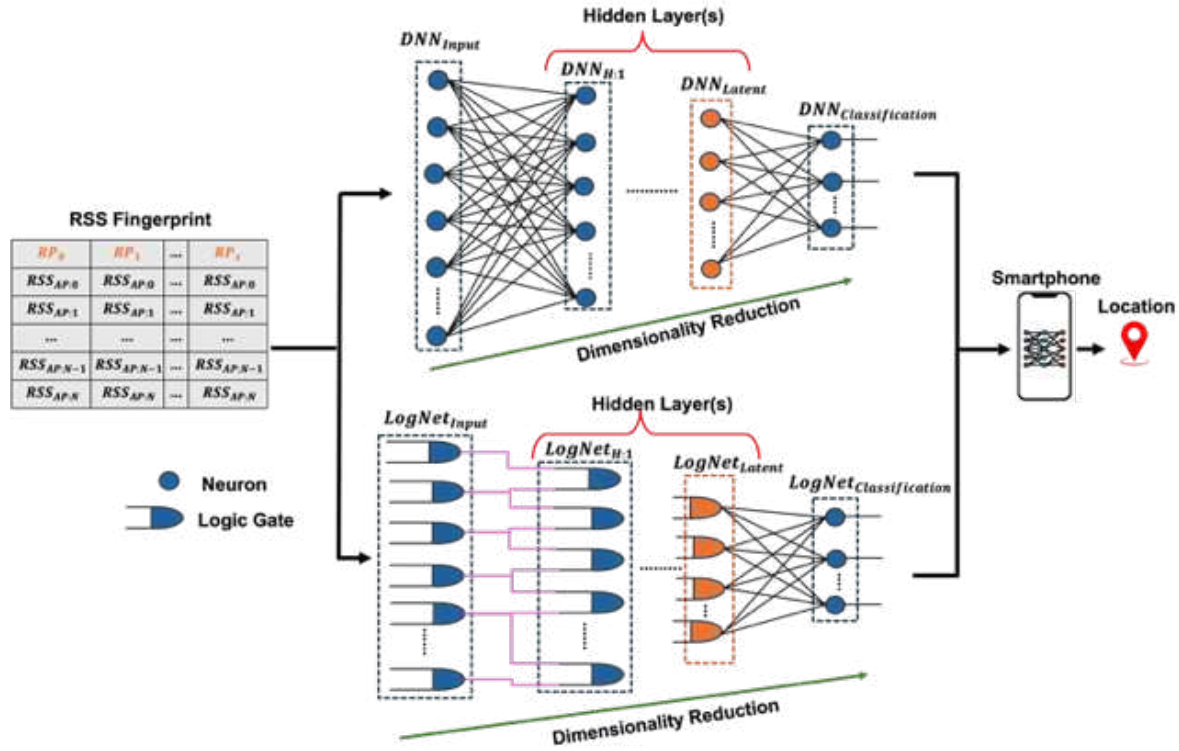


Figure 121: The LogNet and DNN-DownSample architectures.

To design LogNet, we first adopt a baseline DNN-based architecture commonly used in indoor localization, which we term DNN-DownSample. This architecture consists of an input layer (DNN_{Input}) for RSS fingerprints, followed by a sequence of *hidden* layers ($DNN_{H:N}$), where N is the number of layers with progressively decreasing neurons—thereby performing dimensionality reduction—and a final *Softmax* output layer for RP classification, as shown in Figure 121. This DNN-DownSample architecture has been widely used in prior DNN-based indoor localization works, including DNNLOC [78], SANGRIA [82], and AUTLOC [43] due to its effectiveness in compressing high-dimensional RSS vectors and mitigating the effects of short-term temporal

variations. Each *hidden* layer reduces the feature space, often by halving the neuron count from the previous layer, resulting in a compact latent space representation. As shown in Figure 121, each neuron in this architecture computes a weighted sum over its inputs followed by a non-linear activation function:

$$Z = \sigma\left(\sum_i w_i \cdot x_i + B_i\right) \quad (91)$$

where x_i are the input features, σ is the activation function (ReLU), w_i and B_i are the weights and bias of the i^{th} layer respectively. This model is trained end-to-end using backpropagation to minimize the classification loss (*Sparse Categorical Crossentropy*). However, this process results in an opaque latent space, making it difficult to trace how specific APs influence the final indoor localization decisions (shown in Section 13.6.3).

LogNet addresses this limitation by replacing these opaque transformations (neurons) with logic gates that explicitly model pairwise AP interactions. As shown in Figure 121, the process begins by binarizing the RSS fingerprint vectors (X_{train}). All RSS values are first normalized to the $[0, 1]$ range and then a threshold is applied using a fixed value $\emptyset$ (e.g., 0.5). Any normalized RSS above $\emptyset$ is treated as active (1), and values below are considered inactive (0). This yields a binary fingerprint vector (B_{train}) for each training sample. In the first logic layer (L_{input}), the binary RSS values are grouped into adjacent pairs. Each pair is passed through a selected logic gate (G), producing a single binary output (G_{output}). As a result, the output dimension of the first logic layer is half the size of the input (zero-padding applied to maintain even length). The outputs of L_{input} become the inputs to the next layer (L_k), where the same pairwise gate operation is

repeated. This process continues for a predefined number of layers (*hidden*), forming a DNN-DownSample-like hierarchy where each layer compresses the feature space while retaining key activation patterns. The final output after *hidden* defines the LogNet latent space (L_{latent}) —a compact binary encoding that captures discriminative information across APs. L_{latent} is then passed to a fully connected *Softmax* layer, which maps it to the final predicted RP class. *To the best of our knowledge, this is the first work to employ logic gates for interpreting AI models in indoor localization.*

ALGORITHM 1: The LogNet Algorithm

Input: RSS fingerprints X_{train} , RP classes Y_{train} , threshold θ , gate type G , number of logic layers *hidden*

Output: LogNet latent space L_{latent} , Trained LogNet Model M

```

1:  $B_{train} \leftarrow \text{Binarize}(X_{train}, \emptyset) : B[i]=1 \text{ if } X_{train}[i] \geq \emptyset, \text{ else } 0$ 
2:  $L_{input} \leftarrow B_{train}$ 
3: for  $k = 1$  to hidden do
4:   for sample  $SMP \in \text{layer } L_k$  do
5:     if  $\text{len}(SMP) \bmod 2 \neq 0$ 
6:        $SMP \leftarrow \text{append}(SMP, 0)$ 
7:        $Temp \leftarrow []$ 
8:       for  $i = 0$  to  $\text{len}(SMP) - 1$  do
9:          $G_{output} \leftarrow \text{ApplyGate}(SMP[i], SMP[i+1], G)$ 
10:         $Temp.append(G_{output})$ 
11:       $L_{k+1}.append(Temp)$ 
12:     $L_{saved} \leftarrow L_{k+1}$ 
13:  $L_{latent} \leftarrow L_{saved}$ 
14:  $M \leftarrow \text{Train Softmax layer } (L_{latent}, Y_{train})$ 

```

The complete training process—binarization, logic layer construction, and classifier training—is formalized in Algorithm 1. First, RSS fingerprints X_{train} are binarized using threshold $\emptyset$ to produce binarized fingerprints B_{train} , which are then assigned as input to L_{input} (lines 1–2). Next,

the logic gate network is constructed for all *hidden* layers (lines 3-12). In each layer L_k , sample vectors (SMP) are formed by grouping 2 adjacent pairs of inputs: directly from B_{train} when $k = 1$, or from the outputs of the previous layer when $k > 1$ (line 4). Zero-padding is applied to ensure even pairing (lines 5–6). A list $Temp$ is initialized to store G_{output} (line 7), and binary gate operations are applied pairwise (lines 8–10) using the selected gate G . The resulting encoded vector ($Temp$) is added to the next layer L_{k+1} (line 11), which becomes the input to the subsequent logic layer L_{saved} (line 12). After all *hidden* layers are applied, the final output L_{latent} is defined (line 13). Finally, a *Softmax* classifier is trained on L_{latent} with labels Y_{train} (line 14), using backpropagation similar to training DNNs for classification.

13.6. EXPERIMENTAL SETUP

In this section, we outline the experimental setup used to evaluate the LogNet framework. The objectives are: 1) to demonstrate how DNN-based indoor localization models can be interpreted; 2) to empirically address the research questions: **RQ1**, **RQ2**, and **RQ3** from Section 13.2, and 3) to assess the performance improvements that LogNet provides over standard DNNs.

We collected real-world Wi-Fi RSS fingerprints using six smartphones from different manufacturers. The devices are: BLU Vivo 8, HTC U11, Samsung Galaxy S7, LG V20, Motorola Z2, and Oneplus 3. Data was gathered across two buildings—Building 1 and Building 2—with distinct architectural and material characteristics. Building 1 consisted of wood and cement interiors and contained 61 RPs with 164 APs per RP, while Building 2 featured a mix of metal and wooden elements and contained 48 RPs with 218 APs per RP. To ensure realistic conditions, fingerprints were captured during regular work hours without restricting human presence. Each collection instance (CI) included six fingerprints per RP, spanning 10 CIs collected over 2 years.

The first three CIs (CI:0-2) were collected on the same day at different times—morning (8 A.M.), afternoon (3 P.M.), and night (9 P.M.)—to account for short term temporal variations. Subsequent CIs were taken at increasing temporal intervals: after 24 hours, 1 week, 1 month, 3 months, 6 months, 1 year, and 2 years (CI:3–9). Data was split using a 5:1 ratio—five fingerprints per RP for training and one for online testing—ensuring consistency in evaluation. A 1-meter RP spacing was maintained across the path. Additionally, we have open-sourced our data [59] to support reproducibility and research in indoor localization.

With this dataset [59], we proceed to evaluate both DNN-DownSample and LogNet under identical training settings. Both models used a learning rate of 0.01 using the Adam optimizer. DNN-DownSample was trained for 500 epochs due to its fully trainable architecture, while LogNet needed only 150, since its rule-based logic gates generate the latent space without training. Only the final classification layer is trained. In our subsequent experiments, we focus on the DNN-DownSample model as the baseline for comparison since it structurally aligns with LogNet's layered reduction pattern, we also include a study comparing LogNet with prior DNN-based architectures used in indoor localization (Section 13.6.4). DNN-DownSample, like other DNN variants offers limited interpretability, making it difficult to trace how specific APs contribute to decisions. In contrast, LogNet exposes AP-level influence through logic gate activations, enabling us to identify the discriminative features (APs) per RP. This insight not only allows model transparency but also enhances localization robustness under temporal variations.

13.6.1. ROBUSTNESS TO TEMPORAL VARIATIONS: LOGNET VS. DNN

We evaluate the performance of six LogNet variants—each using a fixed logic gate (AND, OR, NAND, NOR, XOR, XNOR)—against the baseline DNN-DownSample architecture, with all

models containing 1 hidden layer. All models are trained on the training five fingerprints per RP from CI:0 and tested on the holdout sixth fingerprint (single fingerprint per RP) from CI:0 (to compare performance without any temporal variations). As shown in Figure 122, all LogNet variants significantly outperform the DNN-DownSample baseline. In particular, LogNet-NOR achieves the best performance (lowest localization errors) with a mean localization error of 2.75 meters, followed by LogNet-NAND at 2.89 meters. In contrast, DNN-DownSample records a higher error of 5.81 meters. Compared to DNN-DownSample, LogNet-AND, OR, NAND, NOR, XOR, and XNOR achieve 1.39 $\times$, 1.48 $\times$, 1.50 $\times$, 1.52 $\times$, 1.08 $\times$, and 1.10 $\times$ lower errors, respectively. These performance gains can be attributed to the logic gate-based architecture of LogNet, where deterministic decision rules filter noise and capture meaningful activation relationships among APs (as discussed in Section 13.4.2).

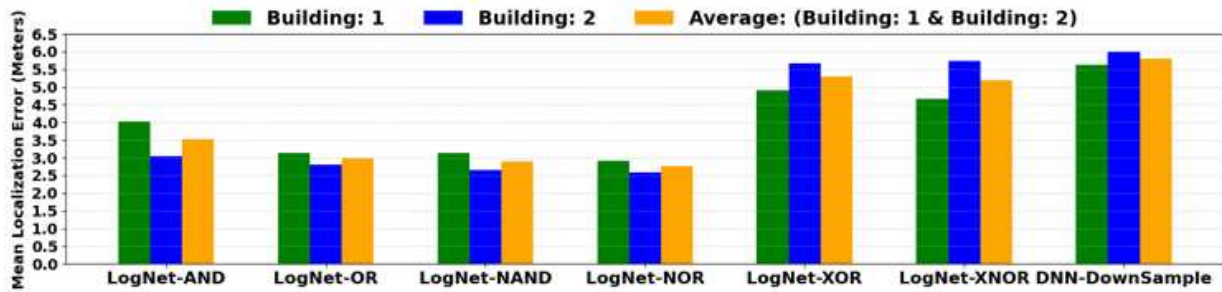


Figure 122: Mean localization error across six LogNet variants and DNN-DownSample at CI:0 (without temporal variations).

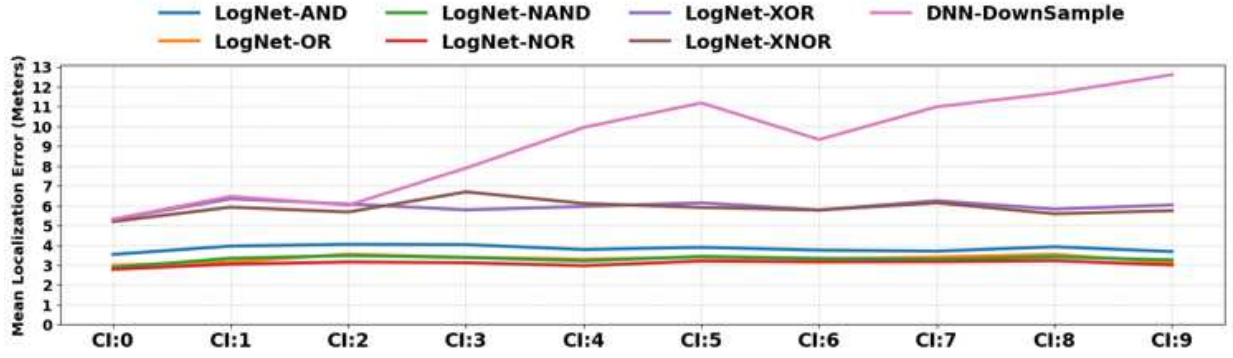


Figure 123: Mean localization error across six LogNet variants and DNN-DownSample over temporal variations (CI:0-9).

To assess temporal robustness (both short and long-term), Figure 123 evaluates all models when trained at CI:0 and tested across CIs:0-9. From CI:0 to CI:2 (collected on the same day), DNN-DownSample performs comparably to LogNet-XOR, indicating that DNNs possess short-term temporal resilience. However, beyond CI:2, DNN error rises steadily peaking at CI:9, showing performance deterioration over long-term temporal variations. This deterioration stems from the DNN’s implicit assumption of ED noise distributions, which fails under non-ED distributions. This distorts the DNN’s decision boundaries in the latent space, leading to unstable predictions. However, all LogNet variants, particularly LogNet-NOR, maintains a flatter error profile with up to $1.76\times$ lower errors than DNN-DownSample. These results directly answer **RQ1**, showing that temporal variation—especially in the form of non-ED noise—corrupts the latent space representations of DNNs, degrading their long-term accuracy. LogNet’s rule-based logic enables it to maintain reliable performance even under long term variations.

13.6.2. LATENT SPACE INTERPRETABILITY AND FAILURE TRACEABILITY

Next, we compare the latent space representations of the best-performing logic gate variant from Section 13.6.2—LogNet-NOR—against the DNN-DownSample model. Both models use a

single hidden layer, resulting in latent space encodings with half the dimensionality of the input RSS fingerprint. For this analysis, we focus on Building 1, which contains 61 RPs and 164 APs per RP, resulting in input vectors of shape (61×164) and latent space encoding of shape (61×82) . Figure 124 visualizes these encodings across all RP classes, where each row corresponds to an RP and each column to its corresponding latent space encodings. As shown in Figure 124, DNN-DownSample produces continuous-valued latent encodings (ranging from 0 to 1), while LogNet generates strict binary encodings (0 or 1). This difference forms the basis for LogNet’s superior interpretability. We highlight two key observations from Figure 124:

- Case 1 (orange bounded boxes):** For RPs 5 and 6, the DNN-DownSample model produces nearly identical and noisy latent space encodings, making it hard to tell the two classes apart or to identify which APs matter. LogNet, however, shows two clear bit differences at bit positions 12 and 46 (highlighted in pink), which separate the RP classes 5 and 6. These bits can be traced back to the APs that most influence the decision. This answers **RQ2**: LogNet helps identify which APs are most influential for distinguishing between classes, allowing us to trace decisions and apply focused fixes. In contrast, the continuous and opaque encodings from DNN-DownSample make this kind of analysis difficult, despite its popularity in indoor localization.
- Case 2 (red bounded boxes):** We now analyze RPs 59 and 60 to understand how non-ED noise affects model predictions. As discussed in Section 13.4, real-world temporal noise is non-ED. This breaks a key assumption that underpins many DNNs, including DNN-DownSample. In DNN-DownSample, we observe small variations at latent bit positions 0, 10, 66, 70, 71, and 77 (we highlight only a subset of bits for brevity, as we observe similar trends across the latent

space) between the two RPs (highlighted in green). The model appears to treat these differences as meaningful. However, these bits are not truly discriminative. If non-ED noise happens to affect these positions (0, 10, 66, 70, 71, and 77) the DNN may misclassify the input, interpreting irrelevant variations as class-defining. Worse, DNN-DownSample offers no way to verify whether these bits reflect true signal or spurious noise, due to its continuous value encoding. In contrast, LogNet exposes the correct structure of the decision boundary. It shows that class differences for RPs 59 and 60 occur at bit positions 6, 41, 42, 49, 61, and 81 (we highlight only a subset of bits for brevity, as we observe similar trends across the latent space; highlighted in blue). These critical bits consistently differ between the two classes. LogNet's rule-based logic gates ignore changes in the unimportant bits unless the noise exceeds a strict threshold, which filters out non-ED noise and focuses only on the APs that truly matter. Because each bit corresponds to a specific AP rule, the model's decisions can be traced to the APs responsible. This answers **RQ3**: LogNet improves robustness by filtering out variations from non-ED noise, and uniquely enables traceability of the most relevant APs for each class decision—features that DNNs lacks.

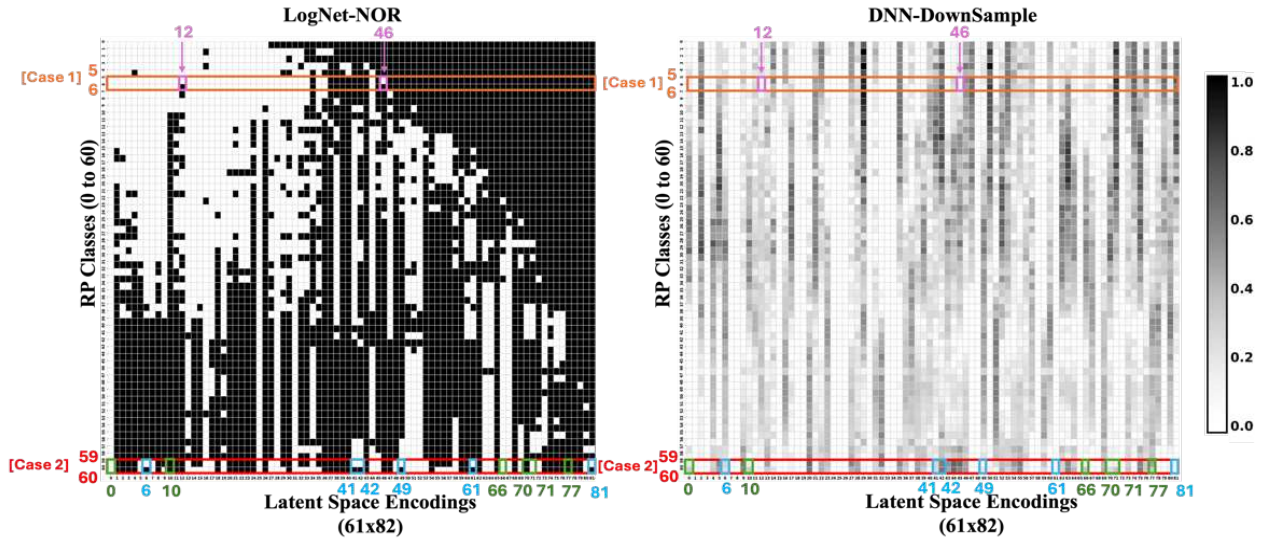


Figure 124: Latent space encodings of LogNet-NOR and DNN-DownSample on Building 1 across all RPs at CI:0.

13.6.3. COMPARISION AGAINST PRIOR WORKS

Next, we evaluate the impact of hidden layer depth on the performance of the best-performing LogNet variant—LogNet-NOR—against the baseline DNN-DownSample model. As shown in Figure 125, models with 1 to 4 hidden layers (1H to 4H) are compared based on mean localization error averaged across all devices, RPs, and CIs (CI:0–CI:9) for both buildings. For DNNs, DNN-DownSample serves as the 1H case, while AUTLOC (2H) [43], SANGRIA (3H) [82], and DNNLOC (4H) [78] represent models from prior works with increasing hidden layers. In Figure 125, the dark black line shows mean localization error, and the whiskers represent best and worst-case performance.

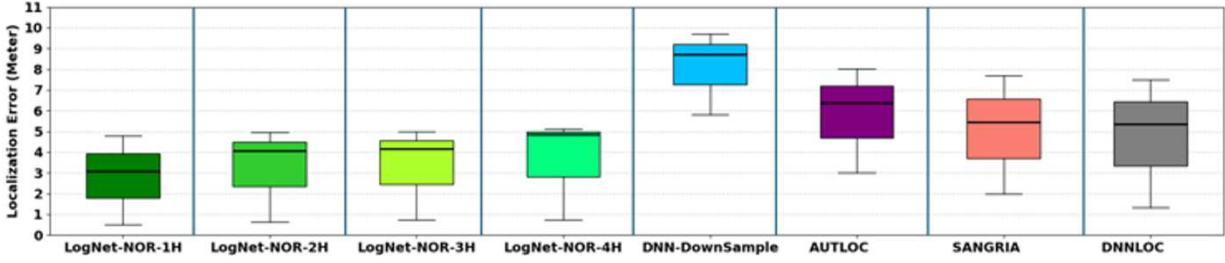


Figure 125: Impact of varying hidden layers across LogNet and DNNs.

Table 22: Comparison of model parameters and model latency.

Model	Total params	Model size (KB)	Model latency (ms)
LogNet-NOR-1H	15374	60.06	601
LogNet-NOR-2H	7871	30.75	577
LogNet-NOR-3H	4028	15.74	567
LogNet-NOR-4H	2198	8.59	393
DNN-DownSample	56027	218.86	934
AUTLOC	66286	228.43	1094
SANGRIA	76546	299.01	1223
DNNLOC	95292	372.2	1451

LogNet-NOR maintains stable mean errors across all depths, due to its aggressive feature compression at each layer, reducing both latent space size and total model parameters, as shown in Table 22 (metrics deployed on smartphones). LogNet-NOR-1H achieves $1.7\times$ to $2.8\times$ lower mean localization errors, $3.6\times$ to $6.1\times$ smaller model size, and $1.5\times$ to $2.4\times$ lower inference latency compared to all DNN variants. Even at 4 hidden layers, LogNet-NOR-4H delivers $1.1\times$ to $1.7\times$ lower errors, $3.4\times$ to $43.3\times$ smaller size, and $2.3\times$ to $3.6\times$ lower latency. These results demonstrate LogNet’s scalability making it highly suitable for mobile deployments.

13.7. CONCLUSIONS

In this study, we introduced LogNet, a novel logic gate-based neural architecture designed to both interpret the decision-making behavior of DNNs in Wi-Fi fingerprinting-based indoor localization and enhance their long-term robustness. By analyzing temporal variation-induced noise, we demonstrated that conventional DNNs fail to capture the true non-Euclidean noise structure observed in real-world environments. LogNet addresses this gap by applying rule-based binary encoding, enabling AP-level traceability and revealing how specific noise patterns disrupt latent decision boundaries. Experimental results confirm that LogNet not only interprets internal model behavior but also delivers significant performance gains—achieving up to $1.1\times$ to $2.8\times$ lower localization error, $3.4\times$ to $43.3\times$ smaller model size, and $1.5\times$ to $3.6\times$ faster inference latency compared to prior DNN-based architectures over 2 years of temporal variations.

CHAPTER 14

DISSERTATION CONCLUSIONS AND FUTURE WORK SUGGESTIONS

1.1. DISSERTATION CONCLUSIONS

This dissertation introduces *SPARK*, a suite of novel and lightweight ML frameworks developed to advance the real-world deployment of Wi-Fi fingerprinting-based indoor localization. As the demand for high-accuracy, energy-efficient, and mobile-friendly localization continues to grow across various domains, *SPARK* directly addresses long-standing technical bottlenecks that have hindered scalable adoption, particularly on resource-constrained mobile devices.

To keep up with this demand, this dissertation systematically targets seven critical challenges. 1) *SPARK* delivers energy-efficient and lightweight ML algorithms optimized for resource-constrained mobile devices, achieving high accuracy without compromising responsiveness or battery life. 2) The *SPARK* frameworks are designed for robustness to device heterogeneity, ensuring consistent performance across a wide range of mobile devices. 3) *SPARK* accounts for temporal variations by supporting stable and long-term indoor localization in environments that evolve over time. 4) Recognizing the cost and privacy concerns of manual data collection, *SPARK* introduces privacy-preserving and efficient data sourcing strategies to alleviate the need for exhaustive site surveys. 5) To handle dynamic and unpredictable indoor environments, *SPARK* offers adaptive learning methods capable of generalizing to new signal distributions without manual recalibration. 6) The *SPARK* frameworks also incorporate security-aware ML models resilient to adversarial threats, thereby ensuring safe deployment in adversarial environments. 7) *SPARK* enhances transparency and interpretability by integrating explainable ML techniques

tailored for indoor localization. These interpretable models provide insight into the ML decision-making process, supporting model debugging, and principled design of future ML architectures.

At the foundation of this dissertation lies a deep investigation into the behavior of Wi-Fi signals in indoor environments, including their susceptibility to real-world noise, signal variability, and adversarial perturbations. However, addressing these challenges requires more than domain knowledge—it demands a rigorous understanding of the underlying ML algorithms themselves. This dissertation carefully examines the behavior, strengths, and limitations of various ML algorithms, particularly under unpredictable input distributions and adversarial conditions. Through this lens, we analyze how ML models internally learn representations, how they generalize across devices, and how they degrade under noise or adversarial interference. This foundational study of both signal dynamics and ML model behavior led to the design of *SPARK* that directly addresses the core open challenges to achieve robust indoor localization.

To initiate this exploration, we introduce ANVIL [45], [135], a novel attention-based neural network designed to learn feature-specific importance within RSS fingerprints. By dynamically assigning higher weights to consistent and robust features, ANVIL [45], [135] exhibits strong resilience to variations introduced by differing hardware configurations. Building on this foundation, we introduce VITAL [80], [136], a novel vision transformer-based architecture that leverages self-attention mechanisms within the transformer model to learn long-range dependencies and global feature representations in RSS vectors. This allows VITAL [80], [136] to further enhance robustness under heterogeneous conditions by capturing more expressive and noise-tolerant representations than conventional attention-based models.

To ensure that ML models remain usable beyond short-term deployments, this dissertation extends the heterogeneity challenge to account for temporal variations—changes in signal

behavior that accumulate over time. To address this, we introduced STELLAR [134], a novel Siamese neural network that applies contrastive learning to model long-term variation in RSS fingerprints. STELLAR [134] learns a latent space representation where fingerprints that are spatially similar, even if collected months or years apart, remain close. This allows the model to remain robust to temporal variations without re-training the model. Unlike conventional heavyweight models that rely on periodic updates, STELLAR [134] was designed for real-world mobile deployment and has been validated across over two years of temporal variation while maintaining both high localization accuracy and a lightweight model footprint.

Recognizing that collecting training data for indoor localization is one of the most significant practical bottlenecks—especially when data must be gathered across a variety of mobile hardware to account for device heterogeneity, and across time to address temporal variations—this dissertation proposes solutions to ease that burden. First, we introduced SANGRIA [82], a novel stacked autoencoder-based ML algorithm designed to generate synthetic RSS fingerprints that closely mimic real-world noise patterns introduced by device heterogeneity. SANGRIA [82] effectively reduces the need for exhaustive, device-specific data collection while maintaining localization accuracy. Building on this, we developed FEDHIL [88], a novel federated learning framework tailored for indoor localization. FEDHIL [88] enables each device to train on its own local data and share only a selectively chosen subset of model weights to a centralized model. This selective aggregation, achieved through a novel federated aggregation algorithm, significantly reduces data collection overhead, improves model robustness, and strengthens user privacy by keeping sensitive data local.

Next, in real-world deployments—particularly in sensitive environments—indoor localization systems are increasingly vulnerable to cyber and adversarial attacks designed to compromise

model accuracy or extract private user data. It is therefore critical to integrate robust security measures directly into the ML framework. To address this, we first introduced CALLOC [87], a novel curriculum adversarial learning strategy that gradually exposes the model to increasingly complex adversarial examples. This enables the model to incrementally adapt and build resilience against a range of potential attacks. Building upon this, we proposed SENTINEL [183], a novel capsule neural network-based framework that learns to identify and suppress manipulated components within adversarial perturbed RSS fingerprints, effectively separating genuine features from attack-induced noise before inference. To extend these protections to distributed settings, we introduced SAFELOC [259], a novel secure federated learning framework that leverages saliency maps to detect and filter out unreliable or potentially malicious weights contributed by local models prior to aggregation. By preventing the propagation of adversarial influence, SAFELOC [259] safeguards both local and global model integrity. All proposed frameworks were evaluated against a spectrum of real and simulated adversarial scenarios—including both targeted and untargeted attacks—and consistently demonstrated significant mitigation of attack impact and enhancing the security of indoor localization systems.

Next, as Indoor localization environments are inherently noisy and highly unpredictable. It is essential for ML models to incorporate learning paradigms that not only remain stable but can also adapt continuously to evolving conditions—something traditional static models fail to achieve. Such adaptability is critical for real-world and long-term deployment. To address this, we developed CIELO [258], a novel class-incremental continual learning framework capable of learning new indoor spaces in real-time. Starting with a subset of known locations, CIELO [258] dynamically incorporates previously unseen indoor spaces during inference, enabling “learning-on-the-go” without retraining from scratch. Recognizing that the noise distribution within each

indoor space also changes unpredictably, we further introduced DAILOC [260], a novel domain-incremental learning method that models and adapts to noise variations in real-time. DAILOC [260] demonstrated resilience over three months of high temporal variation without the need for frequent manual recalibration. To allow context-aware learning, we proposed GATE [261], a novel graph-based neural network that incorporates the spatial geometry of indoor layouts by representing each location as a node in a graph. This topology-aware design allows the model to jointly learn from signal variation and spatial relationships, achieving high localization accuracy while maintaining a lightweight footprint suitable for mobile deployment.

Finally, recognizing that most modern ML algorithms operate as opaque “black box” functions with limited interpretability. In this dissertation, we address a critical limitation that hinders the development of targeted and real-world solutions. Understanding how these models make decisions—and how environmental noise influences those decisions—is essential for both improving robustness and guiding model design. This motivates a shift toward explainable machine learning (XML). In this context, we developed LogNet [262], a novel logic gate-based neural network that emulates the internal decision-making process of traditional ML models. LogNet [262] traces how information flows through the network, identifies the most influential features, and reveals the factors contributing to mispredictions. By exposing these mechanisms, LogNet [262] provides actionable insights for designing models that are better suited to specific indoor environments and ultimately enabling more accurate and resilient indoor localization systems.

All proposed *SPARK* frameworks were deployed and rigorously validated using data collected from diverse real-world indoor environments through multiple commercial smartphones, without introducing any artificial alterations to the signal conditions. Evaluations focused not only on

localization accuracy and robustness but also on inference latency, energy efficiency, and overall feasibility for practical on-device deployment.

1.2. SUGGESTIONS FOR FUTURE WORK

Given the rapid pace of technological advancement and the global shift toward designing more intelligent ML algorithms, there is an increasing need for context-aware, high-accuracy, and lightweight indoor localization frameworks. To remain relevant and effective, future innovations in this domain must evolve to capture real-world complexity with greater fidelity. With this perspective, we identify the following as promising directions for future research.

- **Explore integration of Channel State Information (CSI) and Received Signal Strength (RSS) for enhanced localization accuracy:** While *SPARK* has focused exclusively on RSS, the potential of combining it with CSI remains under explored. Future systems can fuse the coarse-grained, easily accessible measurements from RSS with the fine-grained spatial and frequency-domain details provided by CSI. ML architectures—such as attention-based fusion transformers—could adaptively weigh these complementary modalities based on environmental dynamics, enabling superior robustness and accuracy in complex indoor environments.
- **Multi-modal sensor fusion for enhanced context aware learning:** Beyond relying solely on Wi-Fi fingerprints, future indoor localization frameworks could integrate complementary sensory inputs such as camera imagery, inertial measurement unit (IMU) data, magnetic field maps, and environmental metadata. Leveraging advanced multi-modal fusion techniques,

particularly multi-modal graph-based networks, could be explored to capture relationships between diverse sensor modalities and their spatial correlations. Such architectures can infer not only a user's location but also the surrounding context—such as obstacles, crowd density, or floor layout—enabling adaptive and personalized localization routing that remain robust in dynamic and visually complex environments.

- **Moving from sub-meter localization to centimeter (cm)-level precision:** While current fingerprinting-based approaches often achieve sub-meter accuracy, emerging applications—such as augmented reality (AR), autonomous indoor navigation, and high-precision asset tracking—demand centimeter-level localization. Future indoor localization systems could achieve this by fusing ultra-wideband (UWB) measurements with Wi-Fi fingerprints, leveraging UWB's fine time-of-flight resolution alongside Wi-Fi's spatial coverage. Advanced ML architectures could be designed to intelligently adapt the weighting of each modality in real-time, accounting for changes in signal quality and environmental complexity, to deliver stable centimeter-level accuracy.
- **Path learning and intelligent re-routing (from “where am I?” to “where should I go?”):** Future indoor localization systems should evolve from merely determining a user's position to actively guiding them toward optimal routes. By continuously learning the spatial layout of an environment, ML models could dynamically identify new paths, re-route users to avoid congestion, and tailor navigation to application-specific needs such as accessibility requirements or emergency evacuation priorities. This would involve integrating real-time

localization data with graph-based path planning algorithms, enabling systems to offer not just accurate localization, but also intelligent, context-aware navigation assistance.

- **Multi-floor continual learning (hierarchical building learning):** Indoor localization models are often trained on single-floor data, requiring separate calibration for each new floor. Future systems could adopt continual learning strategies that enable a model trained on one floor to incrementally learn the spatial and signal characteristics of additional floors without forgetting previously acquired knowledge. By structuring the model's understanding hierarchically—capturing both floor-level and building-wide patterns—ML algorithms could be designed to generalize across multi-floor layouts while adapting to unique floor-specific environments. This would reduce the need for complete retraining, support faster deployment in large buildings, and enhance scalability for complex, multi-floor indoor spaces.
- **Preventing ML model corruption during long-term updates:** When indoor localization models are continually updated with large volumes of temporally evolving data, there is a risk of model corruption, where the learned representations degrade, and predictions become unreliable or erratic. Future research should investigate robust continual learning and model maintenance techniques to mitigate catastrophic forgetting and noise accumulation within the model's updates. By ensuring that updates enhance rather than degrade performance, these strategies would enable sustained deployment of indoor localization systems in dynamic and real-world environments.

- **Explore next-generation ML defenses for innovative attacks:** As indoor localization systems become more intelligent and widely deployed, adversaries are likely to develop increasingly sophisticated and unconventional attack strategies targeting both the data and the model. Future research should focus on designing proactive ML defenses that anticipate and neutralize such attacks before they cause significant degradation. This could include adaptive anomaly detection models that continuously learn evolving attack signatures, newer adversarial training pipelines, and innovative ensemble-based defense mechanisms that diversify model decision boundaries. By preparing for creative and unforeseen adversarial strategies, next-generation systems can remain resilient under real-world threat conditions.
- **Explore ML model pruning and quantization for wearable device scale:** To enable seamless deployment of advanced indoor localization frameworks on resource-constrained devices such as smartphones, wearables, and IoT devices, future research should focus on accuracy-preserving model compression techniques. Techniques such as pruning and quantization can be used to reduce model memory footprint without sacrificing accuracy. Exploring such techniques could yield ultra-lightweight models optimized for device-specific hardware. This would not only lower inference latency and power consumption but also expand the reach of high-precision localization to a broader range of consumer devices.
- **Higher explainability for advanced ML architectures in indoor localization:** As ML models for indoor localization grow in complexity, understanding their decision-making processes becomes critical for trust, debugging, and performance optimization. Future systems should integrate explainable ML (XML) techniques to reveal which features, APs, or

environmental cues most influence predictions. Such transparency would enable developers to diagnose performance degradation under environmental changes, facilitate targeted retraining, and provide end-users with interpretable confidence metrics. Ultimately, higher explainability could bridge the gap between raw model accuracy and trustworthy deployment in safety-critical and user-sensitive applications.

BIBLIOGRAPHY

- [1] “Target and Retailers Using Hybrid Indoor Location Tech to Enable IndoorLBS,” 2022 [Online] <http://www.indoorlbs.com/new-http://www.indoorlbs.com/new-blog>
- [2] C. Rodriguez-Martinez and J. Torres-Sospedra, “Revisiting the Analysis of Hyperparameters in k-NN for Wi-Fi and BLE Fingerprinting: Current Status and General Results”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.
- [3] A. Raza, L. Lolic, S. Akhter, and M. Liut, “Comparing and Evaluating Indoor Positioning Techniques”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.
- [4] C. Langlois, S. Tiku, and S. Pasricha, “Indoor localization with smartphones”, *IEEE Consumer Electronics Magazine (CEM)*, 2017.
- [5] S. Pasricha, V. Ugave, Q. Han, and C. Anderson, “LearnLoc: A framework for smart indoor localization with embedded mobile devices”, *IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2015.
- [6] A. Mittal, S. Tiku, and S. Pasricha, “Adapting Convolutional Neural Networks for Indoor Localization with Smart Mobile Devices”, *ACM Great Lakes Symposium on VLSI (GLSVLSI)*, 2018.
- [7] V. Singh, G. Aggarwal, and B. V. S. Ujwal, “Ensemble Based Real-Time Indoor Localization using Stray Wifi Signal”, *IEEE International Conference on Consumer Electronics (ICCE)*, 2018.

- [8] H. Zou, B. Huang, X. Lu, H. Jiang, and L. Xie, “A Robust Indoor Positioning System Based on the Procrustes Analysis and Weighted Extreme Learning Machine”, *IEEE Transactions on Wireless Communications (TWC)*, 2016.
- [9] A. Adege, L. Yen, H.P. Lin, Y. Yayeh, Y. Ruei Li, S.S. Jeng, and G. Berie, “Applying Deep Neural Network (DNN) for large-scale indoor localization using feed-forward neural network (FFNN) algorithm”, *IEEE International Conference on Applied System Innovation (ICASI)*, 2018.
- [10] H. Rizk, “Device-Invariant Cellular-Based Indoor Localization System Using Deep Learning”, *ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2019.
- [11] M. B. Kjaergaard, “Indoor Location Fingerprinting with Heterogeneous Clients”, *Elsevier Pervasive Mobile Computing (PMC)*, 2011.
- [12] C. Figuera, J. L. Rojo-Álvarez, I. Mora-Jiménez, A. Guerrero-Curieses, M. Wilby, and J. Ramos-López, “Time-space sampling and mobile device calibration for WiFi indoor location systems”, *IEEE Transactions on Mobile Computing (TMC)*, 2011.
- [13] S. H. Fang, C. H. Wang, S. M. Chiou, and P. Lin, “Calibration-free approaches for robust Wi-Fi positioning against device diversity: A performance comparison”, *IEEE 75th vehicular technology conference (VTC Spring)*, 2012.
- [14] M. B. Kjærgaard, and C. V. Munk, “Hyperbolic location fingerprinting: A calibration-free solution for handling differences in signal strength (concise contribution)”, *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2008.

- [15] A. M. Hossain, Y. J. Wee-Seng Soh, and H. N. Van. “SSD: A robust RF location fingerprint addressing mobile devices' heterogeneity”, *IEEE Transactions on Mobile Computing (TMC)*, 2013.
- [16] Y. Wei and R. Zheng, “Handling Device Heterogeneity in Wi-Fi based Indoor Positioning Systems”, *IEEE International Conference on Computer Communications (INFOCOM)*, 2020.
- [17] M. Abbas, M. M. Elhamshary, H. Rizk, M. Torki, and M. Youssef, “WiDeep: WiFi-based accurate and robust indoor localization system using deep learning”, *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2019.
- [18] Q. Niu, T. He, N. Liu, S. He, X. Luo, and F. Zhou, “MAIL: Multi-scale attention-guided indoor localization using geomagnetic sequences”, *ACM Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2020.
- [19] M. Abid, P. Compagnon, and G. Lefebvre. “Improved CNN-based magnetic indoor positioning system using attention mechanism”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.
- [20] H. Choi, T. Matsui, S. Misaki, A. Miyaji, M. Fujimoto, and K. Yasumoto, “Simultaneous crowd estimation in counting and localization using WiFi CSI”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.
- [21] S. Fan, S. Y. Wu, C. Han, and X. Wang, “SIABR: A structured intra-attention bidirectional recurrent deep learning method for ultra-accurate terahertz indoor localization”, *IEEE Journal on Selected Areas in Communications (JSAC)*, 2021.

- [22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need", *Advances in neural information processing systems (NeurIPS)*, 2017.
- [23] C. Chen, Q. Fan, and R. Panda, "CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification", *International Conference on Computer Vision (ICCV)*, 2021.
- [24] "Tensorflow Data Augmentation APIs," 2022 [Online]
https://www.tensorflow.org/tutorials/images/data_augmentation
- [25] Y. Jiahui, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention", *IEEE conference on computer vision and pattern recognition (CVPR)*, 2018.
- [26] J. B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layer", *International Conference on Computer Vision (ICLR)*, 2020.
- [27] S. Fang, C. Wang, S. Chiou, and P. Lin, "Calibration-free approaches for robust Wi-Fi positioning against device diversity: A performance comparison", *IEEE 75th vehicular technology conference (VTC Spring)*, 2012.
- [28] S. Tikku, S. Pasricha, B. Notaros, and Q. Han, "CNNLoc: Deep-Learning Based Indoor Localization with WiFi Fingerprinting", *IEEE International Conference on Embedded Software and Systems (ICESS)*, 2019
- [29] S. Tikku, and S. Pasricha, "Overcoming security vulnerabilities in deep learning based indoor localization on mobile devices", *ACM Transactions on Embedded Computing Systems (TECS)*, 2020.

- [30] L. Wang, S. Tiku, and S. Pasricha, "CHISEL: Compression-Aware High-Accuracy Embedded Indoor Localization with Deep Learning", *IEEE Embedded System Letters (ESL)*, 2021.
- [31] S. Tiku, P. Kale, and S. Pasricha, "QuickLoc: Adaptive Deep-Learning for Fast Indoor Localization with Mobile Devices", *ACM Transactions on Cyber-Physical Systems (TCPS)*, 2021.
- [32] S. Tiku, and S. Pasricha, "Siamese Neural Encoders for Long-Term Indoor Localization with Mobile Devices", *IEEE/ACM Design, Automation and Test in Europe (DATE)*, 2022.
- [33] A. H. Salamah, M. Tamazin, M. A. Sharkas, and M. Khedr, "An enhanced WiFi indoor localization system based on machine learning", *IEEE International conference on indoor positioning and indoor navigation (IPIN)*, 2016.
- [34] D. Lymberopoulos and J. Liu, "The Microsoft indoor localization competition: Experiences and lessons learned", *IEEE Signal Processing Society (SPS)*, 2017.
- [35] N. Moayeri, M. O. Ergin, F. Lemic, V. Handziski, and A. Wolisz. "PerfLoc (Part 1): An extensive data repository for development of smartphone indoor localization apps", *IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2016.
- [36] N. Singh, S. Choe, and R. Punmiya, "Machine learning based indoor localization using Wi-Fi RSSI fingerprints: An overview", *IEEE Access*, 2021.
- [37] R. Tian, R. Shen, D. Liu, Y. Wen, and X. Wang, "Performance analysis of RSS fingerprinting based indoor localization", *IEEE Transactions on Mobile Computing (TMC)*, 2017.

- [38] A. B. Adege, L. Yen, H. P. Lin, Y. Yayeh, Y. R. Li, S. S. Jeng, and G. Berie, “Applying Deep Neural Network (DNN) for large-scale indoor localization using feed-forward neural network (FFNN) algorithm”, *IEEE International Conference on Applied System Invention (ICASI)*, 2018.
- [39] Z. Zhongfeng, H. Du, S. Choi, and S. H. Cho, “TIPS: Transformer based indoor positioning system using both CSI and DoA of WiFi signal”, *IEEE Access*, 2022.
- [40] S. Tiku, P. Kale, and S. Pasricha, “QuickLoc: Adaptive Deep-Learning for Fast Indoor Localization with Mobile Devices”, *IEEE Transactions on Industrial Cyber-Physical Systems (TCPS)*, 2021.
- [41] L. Polak, S. Rozum, M. Slanina, T. Bravenec, T. Fryza, and A. Pikrakis, “Received signal strength fingerprinting-based indoor location estimation employing machine learning”, *IEEE Sensors*, 2021.
- [42] W. Njima, A. Bazzi, and M. Chafii, “DNN-based indoor localization under limited dataset using GANs and semi-supervised learning”, *IEEE Access*, 2022.
- [43] J. Liu, N. Liu, Z. Pan, and X. You, “AutLoc: Deep autoencoder for indoor localization with RSS fingerprinting”, *IEEE International Conference on Wireless Communications and Signal Processing (WCSP)*, 2018.
- [44] S. H. Fang C. H. Wang, S. M. Chiou, and P. Lin, “Calibration-free approaches for robust Wi-Fi positioning against device diversity: A performance comparison”, *IEEE 75th vehicular technology conference (VTC Spring)*, 2012.
- [45] S. Tiku, D. Gufran, and S. Pasricha, “Multi-Head Attention Neural Network for Smartphone Invariant Indoor Localization”, *IEEE International conference on indoor positioning and indoor navigation (IPIN)*, 2022.

- [46] S. Tiku, S. Pasricha, B. Notaros and Q. Han, "SHERPA: A Lightweight Smartphone Heterogeneity Resilient Portable Indoor Localization Framework", *IEEE International Conference on Embedded Software and Systems (ICESS)*, 2019.
- [47] S. Tiku, S. Pasricha, "PortLoc: A PorTable Data-driven Indoor Localization Framework for Smartphones", *IEEE Design and Test*, 2019.
- [48] S. Tiku, S. Pasricha, B. Notaros, and Q. Han, "A Hidden Markov Model based Smartphone Heterogeneity Resilient Portable Indoor Localization Framework", *Elsevier Journal of Systems Architecture*, 2020.
- [49] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, and M. Dehghani, "An image is worth 16x16 words: Transformers for image recognition at scale", *IEEE International Conference on Learning Representations (ICLR)*, 2021.
- [50] R. Martinez, C. Cristina, and J. T. Sospedra, "Revisiting the analysis of hyperparameters in k-nn for wi-fi and ble fingerprinting: Current status and general results", *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.
- [51] S. Zhang, W. Xiao, B. Zhang, and B. Hee Soong, "Wireless indoor localization for heterogeneous mobile devices", *IEEE International Conference on Computational Problem-Solving (ICCP)*, 2012.
- [52] A. H. Salamah, M. Tamazin, M. A. Sharkas, and M. Khedr, "An enhanced WiFi indoor localization system based on machine learning", *IEEE International conference on indoor positioning and indoor navigation (IPIN)*, 2016.

- [53] A. Goharfar, J. Babaki, M. Rasti, and P. HJ Nardelli, "Indoor positioning via gradient boosting enhanced with feature augmentation using deep learning", *IEEE Vehicular Technology Conference:(VTC)*, 2022.
- [54] W. Wang, T. Li, W. Wang, and Z. Tu, "Multiple fingerprints-based indoor localization via GBDT: Subspace and RSSI", *IEEE Access*, 2019.
- [55] U. M. Qureshi, Z. Umair, and G. P. Hancke, "Indoor localization using wireless fidelity (WiFi) and bluetooth low energy (BLE) signals", *IEEE International Symposium on Industrial Electronics (ISIE)*, 2019.
- [56] S. Navneet, S. Choe, R. Punmiya, and N. Kaur, "XGBLoc: XGBoost-based indoor localization in multi-building multi-floor environments", *MDPI Sensors*, 2022.
- [57] D. Anna Veronika, V. Ershov, and A. Gulin, "CatBoost: Gradient boosting with categorical features support", *International Conference on Neural Information Processing Systems (NeurIPS)*, 2018.
- [58] "UJIIndoorLoc Dataset", Kaggle, 2014. [Online]. Available: <https://www.kaggle.com/datasets/giantuji/UjiIndoorLoc>.
- [59] EPIC-CSU, "Heterogeneous RSS Indoor Navigation," GitHub, 2021. [Online]. Available: <https://github.com/EPIC-CSU/heterogeneous-rssi-indoor-nav>.
- [60] L. Christopher, S. Tiku, and S. Pasricha, "Indoor Localization with Smartphones: Harnessing the Sensor Suite in Your Pocket", *IEEE Consumer Electronics Magazine CEM*, 2017.
- [61] S. J. Hayward, Kate van Lopik, Christopher Hinde, and A. A. West, "A Survey of Indoor Localization Technologies, Techniques and Applications in Industry," *IEEE Internet of Thing Journal*, 2022.

- [62] E. Martin, O. Vinyals, G. Friedland, and R. Bajcsy, “Precise indoor localization using smart phones”, *ACM International Conference on Mechatronics (ICM)*, 2010.
- [63] F. Zafari, A. Gkelias, and K. K. Leung, “A Survey of Indoor Localization Systems and Technologies,” *IEEE Communications Surveys & Tutorials (CST)*, 2019.
- [64] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, R. Raulefs, and E. Aboutanios, “Recent Advances in Indoor Localization: A Survey on Theoretical Approaches and Applications”, *IEEE Communications Surveys & Tutorials (CST)*, 2017
- [65] S. Shang, and L. Wang, “Overview of WiFi fingerprinting-based indoor positioning”, *IET Communications*, 2022
- [66] X. Feng, K. An Nguyen, and Z. Luo, “A survey of deep learning approaches for WiFi-based indoor positioning,” *Journal of Information and Telecommunication*, 2022.
- [67] J. Song, M. Patel, and M. Ghaffari, “Fusing Convolutional Neural Network and Geometric Constraint for Image-Based Indoor Localization,” *IEEE Robotics and Automation Letters (RAL)*, 2022.
- [68] A. Kumar Panja, S. Fahim Karim, S. Neogy, and C. Chowdhury, “A novel feature based ensemble learning model for indoor localization of smartphone users”, *Elsevier engineering applications of artificial intelligence*, 2022
- [69] A. Khattak, S. Bin, F. Moustafa M. Nasralla, M. Abdullah Esmail, H. Mostafa, and M. Jia, “WLAN RSS-Based Fingerprinting for Indoor Localization: A Machine Learning Inspired Bag-of-Features Approach”, *IEEE Sensors*, 2022.

- [70] P. Roy, and C. Chowdhury, “A survey on ubiquitous WiFi-based indoor localization system for smartphone users from implementation perspectives,” *Springer CCF Transactions on Pervasive Computing and Interaction (TPCI)*, 2022.
- [71] Y. Zheng, J. Liu, M. Sheng, and C. Zhou, “Exploiting Fingerprint Correlation for Fingerprint-Based Indoor Localization: A Deep Learning-Based Approach”, *Springer Nature Machine Learning for Indoor Localization and Navigation*, 2023.
- [72] H. Rizk, T. Amano, H. Yamaguchi, and M. Youssef, “Demonstrating OmniCells: a resilient indoor localization system to devices’ diversity,” *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2022.
- [73] C. Yin, H. Jiang, and J. Chen, “Overview of Approaches for Device Heterogeneity Management During Indoor Localization,” *Springer Nature Machine Learning for Indoor Localization and Navigation*, 2023.
- [74] G. Martín Mendoza-Silva, P. Richter, J. Torres-Sospedra, E. Simona Lohan, and J. Huerta, “Long-Term WiFi Fingerprinting Dataset for Research on Robust Indoor Positioning”, *MDPI Data*, 2018.
- [75] N. Moayeri, M. Onur Ergin, F. Lemic, V. Handziski, and A. Wolisz, “PerfLoc (Part 1): An extensive data repository for development of smartphone indoor localization apps”, *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2016.
- [76] C. Elyca Anak Bundak, M. Amiruddin Abd Rahman, M. Khalis Abd Karim, and N. Huda Osman, “Effect of Different Signal Weighting Function of Magnetic Field Using KNN for Indoor Localization,” *Springer Nature Recent Trends in Mechatronics Towards Industry 4.0.*, 2023.

- [77] Y.S. Chen, C.S. Hsu, and R.S. Chung, “A Semi-Supervised 3D Indoor Localization Using Multi-Kernel Learning for WiFi Networks”, *IEEE Sensors*, 2022.
- [78] B. Jia, Z. Zong, B. Huang, and T. Baker, “A DNN-based WiFi-RSSI Indoor Localization Method in IoT,” *Springer Nature Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST)*, 2021.
- [79] B. Zhao, D. Zhu, T. Xi, C. Jia, S. Jiang, and S. Wang, “Convolutional neural network and dual-factor enhanced variational Bayes adaptive Kalman filter based indoor localization with Wi-Fi”, *Computer Networks*, 2019
- [80] D. Gufran, S. Tiku, and S. Pasricha, “VITAL: Vision Transformer Neural Networks for Accurate Smartphone Heterogeneity Resilient Indoor Localization”, *IEEE/ACM Design Automation Conference (DAC)*, 2023.
- [81] Z. Turgut, and A. Gorgulu Kakisim, “An explainable hybrid deep learning architecture for WiFi-based indoor localization in Internet of Things environment”, *Future Generation Computer Systems*, 2024.
- [82] D. Gufran, S. Tiku, and S. Pasricha, “SANGRIA: Stacked Autoencoder Neural Networks with Gradient Boosting for Indoor Localization,” *IEEE Embedded Systems Letters (ESL)*, 2023.
- [83] R. Montoliu, E. Sansano, O. Belmonte, and J. Torres-Sospedra , “A New Methodology for Long-Term Maintenance of WiFi Fingerprinting Radio Maps”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2018.

- [84] A. Pandey, R. Sequeira, and S. Kumar, “SELE: RSS-Based Siamese Embedding Location Estimator for a Dynamic IoT Environment”, *IEEE Internet of Things (IOT)*, 2022.
- [85] F. Zafari, A. Gkelias, and K. K. Leung, “A Survey of Indoor Localization Systems and Technologies”, *IEEE Communications Surveys & Tutorials (CST)*, 2019.
- [86] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, “Supervised contrastive learning”, *Neural Information Processing Systems (NeurIPS)*, 2020.
- [87] D. Gufran and S. Pasricha, “CALLOC: Curriculum Adversarial Learning for Secure and Robust Indoor Localization”, *IEEE/ACM Design, Automation & Test in Europe (DATE)*, 2024.
- [88] D. Gufran and S. Pasricha, “FedHIL: Heterogeneity Resilient Federated Learning for Robust Indoor Localization with Mobile Devices”, *ACM Transactions on Embedded Computing Systems (TECS)*, 2023.
- [89] S. Tiku, and S. Pasricha, “An Overview of Indoor Localization Techniques”, *IEEE Consumer Electronics Magazine (CEM)*, 2017.
- [90] H. Obeidat, W. Shuaieb, O. Obeidat, and R. A. Alhameed, “A review of indoor localization techniques and wireless technologies”, *Wireless Personal Communications*, 2021.
- [91] T. Yang, A. Cabani, and H. Chafouk, “A survey of recent indoor localization scenarios and methodologies”, *MDPI Sensors*, 2021.
- [92] Y. Li, S. Williams, B. Moran, and A. Kealy, “A probabilistic indoor localization system for heterogeneous devices”, *IEEE Sensors*, 2019.

- [93] L. LiX. Guo, and N. Ansari, "SmartLoc: Smart wireless indoor localization empowered by machine learning", *IEEE Transactions on Industrial Electronics (TIE)* 2019.
- [94] J. Lin, L. Dang, M. Rahouti, and K. Xiong, "Ml attack models: Adversarial attacks and data poisoning attacks", arXiv 2021." *arXiv preprint arXiv:2112.02797*, 2021.
- [95] S. Kaviani, K. J. Han, and I. Sohn, "Adversarial attacks and defenses on AI in medical imaging informatics: A survey", *Elsevier Expert Systems with Applications*, 2022.
- [96] EPIC-CSU, "Heterogeneous RSS Indoor Navigation", GitHub, 2021. [Online]. Available: <https://github.com/EPIC-CSU/heterogeneous-rssi-indoor-nav>.
- [97] R. Nazir, A. A. Laghari, K. Kumar, S. David, and M. Ali, "Survey on wireless network security", *Archives of Computational Methods in Engineering*, 2022.
- [98] D. Man, L. Bing, and J. Lv, "Indoor localization algorithm based on attribute-independent weighted Naive Bayesian", *International Conference on Cyberspace Innovation of Advanced Technologies (CIAT)*, 2020.
- [99] M. Abbas, M. Elhamshary, H. Rizk, M. Torki, and M. Youssef, "WiDeep: WiFi-based accurate and robust indoor localization system using deep learning", *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2019.
- [100] K. Echizenya, K. Kondo, and T. Kitagawa, "Evaluation of the real-time indoor location and motion direction estimation system applying DNN to RSSI Fingerprints of BLE beacons", *IEEE Global Conference on Consumer Electronics (GCCE)*, 2020.
- [101] N. Wafa, I. Ahriz, R. Zayani, M. Terre, and R. Bouallegue, "Deep CNN for indoor localization in IoT-sensor systems", *MDPI Sensors*, 2018.

- [102] D. Gufran, S. Tiku, and S. Pasricha, “VITAL: Vision transformer neural networks for accurate smartphone heterogeneity resilient indoor localization”, *ACM/IEEE Design Automation Conference (DAC)*, 2023.
- [103] Q. Ye, X. Fan, H. Bie, D. Puthal, T. Wu, X. Song, and G. Fang, “SE-Loc: Security-enhanced indoor localization with semi-supervised deep learning”, *IEEE Transactions on Network Science and Engineering (TNSE)*, 2022.
- [104] M. Patil, X. Wang, X. Wang, and S. Mao, “Adversarial attacks on deep learning-based floor classification and indoor localization”, *ACM Workshop on Wireless Security and Machine Learning (WWSML)*, 2021.
- [105] W. Njima, M. Chafii, A. Chorti, M. Shubair, and H. Vincent Poor, “Indoor localization using data augmentation via selective generative adversarial networks”, *IEEE Access*, 2021.
- [106] F. Xiao, Y. Huang, Y. Zuo, W. Kuang, and W. Wang, “Over-the-air adversarial attacks on deep learning Wi-Fi fingerprinting”, *IEEE Internet of Things Journal (IOT)*, 2023.
- [107] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples”, *arXiv preprint arXiv:1412.6572*, 2014.
- [108] S. Finlayson, H. Won Chung, I. S. Kohane, and A. L. Beam, “Adversarial attacks against medical deep learning systems”, *arXiv preprint arXiv:1804.05296*, 2018.
- [109] Y. Dong F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting adversarial attacks with momentum”, *IEEE conference on computer vision and pattern recognition (CVPR)*, 2018.

- [110] A. Petrenko, A. Sizo, W. Qian, AD. Knowles, A. Tavassolian, K. Stanley, S. Bell, “Exploring mobility indoors: an application of sensor-based and GIS systems”, *Transactions in GIS*, 2014.
- [111] WK. Zegeye, SB. Amsalu, F. Moazzami, “WiFi RSS fingerprinting indoor localization for mobile devices”, *IEEE Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2016.
- [112] M. Shu, G. Chen, Z. Zhang, “Efficient image-based indoor localization with MEMS aid on the mobile device”, *IEEE International Society for Photogrammetry and Remote Sensing (ISPRS)*, 2022.
- [113] B. Yang, L. Guo, R. Guo, M. Zhao, T. Zhao, “A novel trilateration algorithm for RSSI-based indoor localization”, *IEEE Sensors*, 2020.
- [114] X. Hou, T. Arslan, A. Juri, F. Wang, “Indoor localization for bluetooth low energy devices using weighted off-set triangulation algorithm”, *Institute of Navigation Global Navigation Satellite System (ION GNSS)*, 2016.
- [115] F. Alhomayani, and MH. Mahoor, “Deep learning methods for fingerprint-based indoor positioning: A review”, *Journal of Location Based Services*, 2020.
- [116] D. Duong, Y. Xu, and K. David, “The influence of fast fading and device heterogeneity on wi-fi fingerprinting”, *IEEE Conference on Vehicular Technology (VTC)*, 2018.
- [117] I. Alshami, N. Ahmad, and S. Sahibuddin, “RSS Certainty: An Efficient Solution for RSS Variation due to Device Heterogeneity in WLAN Fingerprinting-based Indoor Positioning System”, *IEEE Palestinian International Conference on Information and Communication Technology (PICICT)*, 2021.

- [118] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions", *IEEE Computer Vision and Pattern Recognition Conference (CVPR)*, 2015.
- [119] W. Xue, X. Hua, Q. Li, W. Qiu, and X. Peng, "Improved clustering algorithm of neighboring reference points based on KNN for indoor localization", *IEEE Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, 2018.
- [120] J. Jadidi, M. Patel, J.V. Miro, "Gaussian processes online observation classification for RSSI-based low-cost indoor positioning systems", *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.
- [121] AB. Adege, HP. Lin, GB. Tarekegn, and SS. Jeng, "Applying deep neural network (DNN) for robust indoor localization in multi-building environment", *MDPI Applied Sciences*, 2018.
- [122] M. Dakkak, B. Daachi, A. Nakib, and P. Siarry, "Multi-layer perceptron neural network and nearest neighbor approaches for indoor localization", *IEEE Systems, Man, and Cybernetics Society (SMCS)*, 2014.
- [123] W. Liu, H. Chen, and Q. Cheng, "LC-DNN: Local Connection Based Deep Neural Network for Indoor Localization With CSI", *IEEE Access*, 2020.
- [124] Y. Kim, H. Shin, Y. Chon, and H. Cha, "Smartphone-based Wi-Fi tracking system exploiting the RSS peak to overcome the RSS variance problem", *Elsevier Pervasive and Mobile Computing (PMC)*, 2013.

- [125] G. Elsayed, P. Ramachandran, J. Shlens, and S. Kornblith, “Revisiting spatial invariance with low-rank local connectivity”, *Proceedings of Machine Learning Research (PMLR)*, 2020.
- [126] O. Parkhi, A. Ved, and A. Zisserman, “Deep face recognition”, *British Machine Vision Conference (BMVC)*, 2015.
- [127] Q. Ye, H. Bie, KC. Li, X. Fan, and G. Fang, “Edgeloc: A robust and real-time localization system toward heterogeneous iot devices”, *IEEE Internet of Things (IOT)*, 2021.
- [128] A. Shafahi, M. Najibi, MA. Ghiasi, Z. Xu, LS. Davis, G. Taylor, and T. Goldstein, “Adversarial training for free!”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.
- [129] A. Kurakin, I. Goodfellow, S. Bengio, “Adversarial machine learning at scale”, *arXiv:1611.01236*, 2016.
- [130] X. Wang, X. Wang, S. Mao, J. Zhang, SCG. Periaswamy, J. Patton, “Adversarial deep learning for indoor localization with channel state information tensors”, *IEEE Internet of Things (IOT)*, 2022.
- [131] Q. Lu, H. Qu, Y. Zhuang, X. Lin, Y. Zhu, and Y. Liu, “A Passive Client-based Approach to Detect Evil Twin Attacks”, *IEEE International Conference on Embedded Software and Systems (ICESS)*, 2017.
- [132] A. Mallik, “Man-in-the-middle-attack: Understanding in simple words”, *Cyberspace*, 2019.
- [133] S. Sabour, N. Frosst, and GE. Hinton, “Dynamic routing between capsules”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

- [134] D. Gufran, S. Tiku, and S. Pasricha, “STELLAR: Siamese multiheaded attention neural networks for overcoming temporal variations and device heterogeneity with indoor localization”, *IEEE Journal of Indoor and Seamless Positioning and Navigation (J-ISPIN)*, 2023.
- [135] S. Tiku, D. Gufran, and S. Pasricha, “Smartphone invariant indoor localization using multi-head attention neural network”, *Springer Nature Machine Learning for Indoor Localization and Navigation*, 2023.
- [136] D. Gufran, S. Tiku, and S. Pasricha, “Heterogeneous device resilient indoor localization using vision transformer neural networks”, *Springer Nature Machine Learning for Indoor Localization and Navigation*, 2023.
- [137] S. Sadowski, and P. Spachos, “RSSI-Based Indoor Localization With the Internet of Things”, *IEEE Access*, 2018.
- [138] S. Piao, Z. Ba, and L. Su, “Automating csi measurement with uavs: from problem formulation to energy-optimal solution”, *IEEE International Conference on Computer Communications (INFOCOM)*, 2025.
- [139] Y. Li, Z. He, Z. Gao, Y. Zhuang, and C. Sheimy, “Toward robust crowdsourcing-based localization: A fingerprinting accuracy indicator enhanced wireless/magnetic/inertial integration approach”, *IEEE Internet of Things (IoT)*, 2018.
- [140] P. Zhang, Q., Zhao, Y. Li, X. Ni, and Y. Zhuang, “Collaborative wifi fingerprinting using sensor-based navigation on smartphones, *IEEE Sensors*, 2015.
- [141] J. Xiao, Y. Yi, L. Wang, H. Li, and Z. Zhou, “Nomloc: Calibration-free indoor localization with nomadic access points”, *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2014.

- [142] A. Bose, and F. Chuan, “A practical path loss model for indoor wifi positioning enhancement”, *IEEE International Conference on Information and Communication Systems (ICICS)*, 2007.
- [143] A. Basiri, E. Lohan, T. Moore, and A. Winstanley, “Indoor location-based services challenges, requirements and usability of current solutions”, *Computer Science Review*, 2017.
- [144] X. Shixiong, Y. Liu, G. Yuan, M. Zhu, Z. Wang, “Indoor Fingerprint Positioning Based on Wi-Fi: An Overview”, *International Society for Photogrammetry and Remote Sensing (ISPRS)*, 2017.
- [145] 2022. Target rolls out bluetooth beacon technology in stores to power new indoor maps in its app, [Online] <http://tcn.ch/2fbIM0P>.
- [146] S. Shang, and L. Wang, “Overview of WiFi fingerprinting-based indoor positioning”, *IET Communications*, 2022.
- [147] P. Eberechukwu, H. Park, and C. Laoudias, “DNN-based Indoor Fingerprinting Localization with WiFi FTM”, *IEEE International Conference on Mobile Data Management (MDM)*, 2022.
- [148] G. Liu, Z. Qian, and X. Wang, “An Indoor WLAN Location Algorithm Based on Fingerprint Database Processing”, *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 2020.
- [149] M. Luo, J. Zheng, W. Sun, X. Zhang, “WiFi-based Indoor Localization Using Clustering and Fusion Fingerprint”, *IEEE Chinese Control Conference (CCC)*, 2021.
- [150] N. Nagia, M. Rahman, and S. Valaee, “Federated Learning for WiFi Fingerprinting”, *IEEE Chinese Control Conference (CCC)*, 2022.

- [151] O. Tasbaz, and V. Moghtadaiee, “Zone-Based Federated Learning in Indoor Positioning”, *IEEE International Conference on Computer and Knowledge Engineering (ICCKE)*, 2022.
- [152] M. Jeong, and S. Choi, “A tutorial on Federated Learning methodology for indoor localization with non-IID fingerprint databases”, *ICT Express*, 2023.
- [153] S. Sadowski, and P. Spachos, “Memoryless Techniques and Wireless Technologies for Indoor Localization With the IoT”, *IEEE Internet of Things (IoT)*, 2022.
- [154] Z. Mabunga, and J. Cruz, “Utilization of Different Wireless Technologies’ RSS for Indoor Environment Classification Using SVM”, *IEEE Control and System Graduate Research Colloquium (ICSGRC)*, 2021.
- [155] L. Lee, J. Kim, and N. Moon, “Random Forest and WiFi fingerprint-based indoor location recognition system using smart watch”, *HCCIS*, 2019.
- [156] A. Li, J. Fu, H. Shen, and S. Sun, “A Cluster-Principal-Component-Analysis-Based Indoor Positioning Algorithm”, *IEEE Internet of Things (IoT)*, 2021.
- [157] B. Pinto, R. Barreto, E. Souto, and H. Oliveira, “Robust RSSI-Based Indoor Positioning System Using K-Means Clustering and Bayesian Estimation”, *IEEE Sensors*, 2021.
- [158] J. Yang, H. Zou, H. Jiang, L. Xie, “Device-free Occupant Activity Sensing using WiFi-enabled IoT Devices for Smart Homes”, *IEEE Internet of Things (IoT)*, 2018.
- [159] O. Gokalp, J. Lee, D. Burghal, and A. Molisch, “ Simple and Effective Augmentation Methods for CSI Based Indoor Localization”, *ArXiv:2211.10790v1*, 2022.

- [160] Z. Tang, S. Li, K. Kim, and J. Smith, “Multi-Output Gaussian Process-Based Data Augmentation for Multi-Building and Multi-Floor Indoor Localization”, *IEEE International Conference on Communications (ICC)*, 2022.
- [161] Y. Dong, T. Arslan, and Y. Yang, “A WiFi Fingerprint Augmentation Method for 3-D Crowdsourced Indoor Positioning Systems”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2022.
- [162] X. Chen, H. Li, C. Zhou, X. Liu, and D. Wu, “Fidora: Robust WiFi-Based Indoor Localization via Unsupervised Domain Adaptation”, *IEEE Internet of Things (IoT)*, 2022.
- [163] S. Tiku, D. Gufran, and S. Pasricha, “Smartphone Invariant Indoor Localization Using Multi-head Attention Neural Network”, *Springer Nature Machine Learning for Indoor Localization and Navigation*, 2023.
- [164] W. Li, C. Zhang, and Y. Tanaka, “Pseudo Label-Driven Federated Learning-Based Decentralized Indoor Localization via Mobile Crowdsourcing”, *IEEE Sensors*, 2020.
- [165] B. Gao, F. Yang, N. Cui, K. Xiong, and Y. Lu, “A Federated Learning Framework for Fingerprinting-Based Indoor Localization in Multibuilding and Multifloor Environments”, *IEEE Internet of Things (IoT)*, 2023.
- [166] X. Cheng, C. Ma, J. Li, H. Song, and F. Shu, “Federated Learning-Based Localization With Heterogeneous Fingerprint Database”, *IEEE Wireless Communications Letters (WCL)*, 2022.
- [167] H. Brendan, E. Moore, and D. Ramage, “Communication-Efficient Learning of Deep Networks from Decentralized Data”, *ArXiv.1602.05629*, 2016.

- [168] J. Konečný, H. McMahan, H. Yu, and P. Richtárik, “Federated Learning: Strategies for Improving Communication Efficiency”, *ArXiv.1610.05492*, 2016.
- [169] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both Weights and Connections for Efficient Neural Networks”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2016.
- [170] A. Billa, I. Shayea, A. Alhammadi, Q. Abdullah, and M. Roslee, “An overview of indoor localization technologies: Toward IoT navigation services”, *IEEE International Symposium on Telecommunication Technologies (ISTT)*, 2020.
- [171] MarketsandMarkets. “Indoor Location Market by Component, Technology, Application, Vertical and Region - Global Forecast 2028”, 2025.
- [172] S. Pasricha, “AI and Machine Learning Driven Indoor Localization and Navigation with Mobile Embedded Systems”, *ArXiv:2408.0479*, 2024.
- [173] B. S. Ciftler, A. Albaseer, N. Lasla, and M. Abdallah, “Federated learning for localization: A privacy-preserving crowdsourcing method”, *ArXiv:2001.01911*, 2020.
- [174] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, “Data poisoning attacks on federated machine learning”, *IEEE Internet of Things (IOT)*, 2021.
- [175] F. Yin, Z. Lin, Q. Kong, Y. Xu, D. Li, S. Theodoridis, and SR. Cui, “FedLoc: Federated learning framework for data-driven cooperative localization and location data processing”, *IEEE Open Journal of Signal Processing (JSP)*, 2020.
- [176] T. Sun, D. Li, and B. Wang, “Decentralized federated averaging”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022.
- [177] Y. Etiabi, and E.M. Amhoud, “Federated distillation based indoor localization for IoT networks”, *IEEE Sensors*, 2024.

- [178] A. Hilal, I. Arai, and S. El-Tawab, “DataLoc+: A data augmentation technique for machine learning in room-level indoor localization”, *IEEE Wireless Communications and Networking Conference (WCNC)*, 2021.
- [179] E. Mhamdi, R. Guerraoui, and S. Rouault, “The Hidden Vulnerability of Distributed Learning in Byzantium”, *ArXiv:1802.07927*, 2018.
- [180] H. Jeong, H. Son, S. Lee, J. Hyun, and T.M. Chung, “FedCC: Robust federated learning against model poisoning attacks”, *ArXiv:2212.01976*, 2022.
- [181] T. D. Luong, V.M. Tien, P.T. Duy, and V.H. Pham, “FedLS: An Anti-poisoning Attack Mechanism for Federated Network Intrusion Detection Systems Using Autoencoder-Based Latent Space Representations”, *ISPE*, 2023.
- [182] M. Tsukada, M. Kondo, and H. Matsutani, “A neural network-based on-device learning anomaly detector for edge devices”, *IEEE Transactions on Computers (TC)*, 2020.
- [183] D. Gufran, P. Anandathirtha, and S. Pasricha, “SENTINEL: Securing Indoor Localization against Adversarial Attacks with Capsule Neural Networks”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2024.
- [184] S. Pasricha, “AI-Driven Indoor Navigation with Mobile Embedded Systems”, *IEEE/ACM CODES+ ISSS*, 2024.
- [185] Indoor Location Market by Component (Hardware, Services, Software), Technology (Bluetooth Low Energy (BLE), Infrared, Magnetic Positioning), Organization Size, Application, Deployment, End-User - Global Forecast 2025-2030, [Online]:

<https://www.giiresearch.com/report/ires1580990-indoor-location-market-by-component-hardware.html>

- [186] F. Zafari, A. Gkelias, and K. K. Leung, “A survey of indoor localization systems and technologies”, *IEEE Communications Surveys and Tutorials (CST)*, 2019.
- [187] Y. You, and C. Wu, “Indoor Positioning System with Cellular Network Assistance Based on Received Signal Strength Indication of Beacon”, *IEEE Access*, 2020.
- [188] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, and C. Yuen, “Recent advances in indoor localization: A survey on theoretical approaches and applications”, *IEEE Communications Surveys and Tutorials (CST)*, 2016.
- [189] Z. Yang, Z. Zhou, and Liu, Y, “From RSS to CSI: Indoor localization via channel response”, *ACM Computing Surveys (CSUR)*, 2013.
- [190] G. Pecoraro, S. Di Domenico, E. Cianca, and M. De Sanctis, “CSI-based fingerprinting for indoor localization using LTE signals”, *EURASIP Journal on Advances in Signal Processing*, 2018.
- [191] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez, “Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges”, *Information fusion*, 2020.
- [192] S. Mittal, S. Galess, and T. Brox, “Essentials for class incremental learning”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [193] G. M. van de Ven, T. Tuytelaars, and A.S. Tolias, “Three types of incremental learning”, *Nature Machine Intelligence*, 2022.
- [194] D. R. Ashley, S. Ghiassian, and R.S. Sutton, “Does the Adam Optimizer Exacerbate Catastrophic Forgetting? “, *ArXiv:2102.07686*, 2021.

- [195] R. Coop, and I. Arel, “Mitigation of catastrophic forgetting in recurrent neural networks using a Fixed Expansion Layer”, *International Joint Conference on Neural Networks (IJCNN)*, 2013.
- [196] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, AA. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, “Overcoming catastrophic forgetting in neural networks”, *National Academy of Sciences*, 2017.
- [197] M. De Lange, A. Rahaf, M. Marc, P. Sarah, J. Xu, L. Aleš, S. Gregory, and T. Tinne, “A Continual Learning Survey: Defying Forgetting in Classification Tasks”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022.
- [198] Z. Li, and D. Hoiem, “Learning without forgetting”, *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 2017.
- [199] F. Potorti, S. Park, A. Crivello, F. Palumbo, M. Girolami, P. Barsocchi, S. Lee, J. Torres-Sospedra, ARJ. Ruiz, A. Perez-Navarro, and GM. Mendoza-Silva, “The IPIN 2019 indoor localisation competition—Description and results”, *IEEE Access*, 2020.
- [200] SA. Rebuffi, A. Kolesnikov, G. Sperl, and CH. Lampert, “icarl: Incremental classifier and representation learning”, *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [201] G. Chen, X. Guo, K. Liu, X. Li, and J. Yang, “RWKNN: A modified WKNN algorithm specific for the indoor localization problem”, *IEEE Sensors*, 2022.
- [202] MT. Le, “Enhanced indoor localization based BLE using Gaussian process regression and improved weighted kNN”, *IEEE Access*, 2021.

- [203] PS. Varma, and V. Anand, “Indoor localization for IoT applications: Review, challenges and manual site survey approach”, *IEEE Bombay Section Signature Conference (IBSSC)*, 2021.
- [204] P. Blanchard, EME. Mhamdi, R. Guerraoui, and J. Stainer, “Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.
- [205] A.A. Rusu, N.C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, “Progressive neural networks”, *ArXiv:1606.04671*, 2016.
- [206] H. Shin, J. K. Lee, J. Kim, and J. Kim, “Continual learning with deep generative replay”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.
- [207] J. Xu, and Z. Zhanxing, “Reinforced continual learning”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.
- [208] C. Lamers, R. Vidal, N. Belbachir, N. van Stein, and T. Bäeck, P. Giampouras, “Clustering-based domain-incremental learning”, *IEEE International Conference on Computer Vision (CVF)*, 2023.
- [209] P. Garg, R. Saluja, VN. Balasubramanian, C. Arora, A. Subramanian, and CV. Jawahar, “Multi-domain incremental learning for semantic segmentation”, *IEEE Conference on Applications of Computer Vision (CVF)*, 2022.
- [210] G. Oren, and L. Wolf, “In defense of the learning without forgetting for task incremental learning”, *IEEE International Conference on Computer Vision (CVF)*, 2021.

- [211] DW. Zhou, QW. Wang, ZH. Qi, HJ. Ye, DC. Zhan, and Z. Liu, “Class-incremental learning: A survey”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2024.
- [212] A. Abolfazli, and E. Ntoutsis, “Drift-aware multi-memory model for imbalanced data streams”, *IEEE International Conference on Big Data (Big Data)*, 2020.
- [213] Indoor Location Market by Component, Technology, Organization Size, Application, Deployment, End-User - Global Forecast 2025-2030, [Online]: <https://www.giiresearch.com/report/ires1580990-indoor-location-market-by-component-hardware.html>
- [214] X. Feng, K. A. Nguyen, and Z. Luo. “A dynamic model switching algorithm for WiFi fingerprinting indoor positioning” *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2023.
- [215] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, “A survey on federated learning: The journey from centralized to distributed on-site learning and beyond”, *IEEE Internet of Things Journal (IOT)*, 2021.
- [216] Z. Li, B. Ko, and H. Jin Choi, “Naive semi-supervised deep learning using pseudo-label”, *Peer-to-peer networking and applications*, 2019.
- [217] Z. Jia, and B. Guan, “Received signal strength difference–based tracking estimation method for arbitrarily moving target in wireless sensor networks”, *International Journal of Distributed Sensor Networks*, 2018.
- [218] Li, Hang, X. Chen, J. Wang, D. Wu, and X. Liu, “DAFI: WiFi-based device-free indoor localization via domain adaptation”, *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2021.

- [219] L. Li, X. Guo, M. Zhao, H. Li, and N. Ansari, "TransLoc: A heterogeneous knowledge transfer framework for fingerprint-based indoor localization", *IEEE Transactions on Wireless Communications*, 2021.
- [220] M. Mirza, J. Marc Masana, H. Possegger, and H. Bischof, "An efficient domain-incremental learning approach to drive in all weather conditions", *IEEE conference on computer vision and pattern recognition (CVF)*, 2022.
- [221] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application", *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, 2024.
- [222] K. Jeeveswaran, E. Arani, and B. Zonooz, "Gradual divergence for seamless adaptation: A novel domain incremental learning method", *ArXiv preprint arXiv:2406.16231*, 2024.
- [223] D. Bouchacourt R. Tomioka, and S. Nowozin, "Multi-level variational autoencoder: Learning disentangled representations from grouped observations", *AAAI Conference on Artificial Intelligence*, 2018.
- [224] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The active badge location system", *ACM Transactions on Information Systems (TOIS)*, 1992.
- [225] MarketsandMarkets. "Indoor Location Market by Component, Technology, Application, Vertical and Region - Global Forecast to 2028." 2024 [Online], <https://www.marketsandmarkets.com/Market-Reports/indoor-location-market-989.html>.

- [226] S. Dai, L. He, and X. Zhang, “Autonomous WiFi fingerprinting for indoor localization”, *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2020.
- [227] S. Asaad, and H. S. Maghdid, “A comprehensive review of indoor/outdoor localization solutions in IoT era: Research challenges and future perspectives”, *Computer Networks*, 2022.
- [228] R. S. Naser, M. C. Lam, F. Qamar, and B. B. Zaidan, “Smartphone-based indoor localization systems: A systematic literature review”, *MDPI Electronics*, 2023.
- [229] X. Zheng, R. Cheng, and Y. Wang, “Rssi-knn: A RSS indoor localization approach with knn”, *IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, 2023.
- [230] M. W. P. Maduranga, V. Tilwari, and R. Abeysekera, “Improved RSS indoor localization in IoT systems with machine learning algorithms”, *Signals*, 2023.
- [231] X. Kang, X. Liang, and Q. Liang, “Indoor localization algorithm based on a high-order graph neural network”, *MDPI Sensors*, 2023.
- [232] S. Wang, S. Zhang, J. Ma, and O. A. Dobre, “Graph Neural Network-Based WiFi Indoor Localization System”, *IEEE Global Communications Conference (GLOBECOM)*, 2024.
- [233] F. Lezama, G. G. González, F. Larroca, and G. Capdehourat, “Indoor localization using graph neural networks”, *IEEE URUCON*, 2021.
- [234] N. Asif, Y. Sarker, R. K. Chakraborty, M. J. Ryan, M. H. Ahamed, D. K. Saha, and F. R. Badal, “Graph neural network: A comprehensive review on non-euclidean space”, *IEEE Access*, 2021.

- [235] L. Chanama, and O. Wongwirat, “A comparison of decision tree based techniques for indoor positioning system”, *IEEE International Conference on Information Networking (ICOIN)*, 2018.
- [236] X. Kang, X. Liang, and Q. Liang, “Indoor localization algorithm based on a high-order graph neural network”, *IEEE Sensors*, 2023.
- [237] X. Luo, and N. Meratnia, “A geometric deep learning framework for accurate indoor localization”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2022.
- [238] R. Vishwakarma, R. B. Joshi, and S. Mishra, “IndoorGNN: A graph neural network based approach for indoor localization using WiFi RSSI”, *Big Data Analytics*, 2023.
- [239] D. Kim, and Y. Suh, “GConvLoc: WiFi fingerprinting-based indoor localization using graph convolutional networks” *IEEE IEICE*, 2023.
- [240] T. He, Q. Niu, and N. Liu, “GC-Loc: A graph attention based framework for collaborative indoor Localization using infrastructure-free signals”, *ACM Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2023.
- [241] U. Alon, and E. Yahav, “On the bottleneck of graph neural networks and its practical implications”, *ArXiv:2006.05205*, 2020.
- [242] Q. Li, Z. Han, and X. M. Wu, “Deeper insights into graph convolutional networks for semi-supervised learning”, *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018.
- [243] J. Lee, V. Thost, B. Kim, J. Kang, and T. Ma, “Understanding and Tackling Over-Dilution in Graph Neural Networks”, *International Conference on Learning Representations (ICLR)*, 2024.

- [244] L. Zhang, Y. Qiu, S. Wu, T. Zhang, and Q. Zhang, “GraphLoc: Enhancing Fingerprint-Based Localization With Graph Representation Learning”, *IEEE Internet of Things (IOT)*, 2025.
- [245] S. Pasricha, “AI and Machine Learning Driven Indoor Localization and Navigation with Embedded Systems”, *ArXiv:2408.04797*, 2024.
- [246] [Online] "<https://www.marketsandmarkets.com/Market-Reports/indoor-location-market-989.html>" *MarketsandMarkets*, 2024.
- [247] K. Ibrahim, F. Yıldırım Okay, O. Muyanlı, and S. Özdemir, “Explainable artificial intelligence (xai) for internet of things: a survey”, *IEEE Internet of Things (IOT)*, 2023.
- [248] H. Jung, J. Yoon, and H. Lee, “eXLoc: Understanding Deep Learning-driven Indoor Localization with eXplainable AI”, *IEEE Global Communications Conference (GLOBECOM)*, 2024.
- [249] P. Wu, S. Su, Z. Zuo, X. Guo, B. Sun, and X. Wen, “Time difference of arrival (TDoA) localization combining weighted least squares and firefly algorithm”, *MDPI Sensors*, 2019.
- [250] M. Heydariaan H. Dabirian, and O. Gnawali, “Anguloc: Concurrent angle of arrival estimation for indoor localization with uwb radios”, *IEEE Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2020.
- [251] S. Azghadi, A. Nuh Mih, A. Kawnine, M. Wachowicz, F. Palma, and H. Cao, “An Adaptive Indoor Localization Approach Using WiFi RSSI Fingerprinting with SLAM-Enabled Robotic Platform and Deep Neural Networks”, *IEEE International*

- Conference on Collaborative Advances in Software and COmputiNg (CASCON)*, 2024.
- [252] B. Stojkoska, and N. Jofce, “Simple Iterative trilateration-based scheme for indoor localization of IoT devices” *ICT Innovations*, 2017.
 - [253] P. Bahl, and V. N. Padmanabhan, “RADAR: An in-building RF-based user location and tracking system”, *IEEE Conference on computer communications (INFOCOM)*, 2000.
 - [254] D. Mao, H. Lin, and X. Lou, “Fingerprinting indoor positioning based on improved sequential deep learning”, *Algorithms*, 2025.
 - [255] Z. LiuB. Dai, X. Wan, and X. Li, “Hybrid wireless fingerprint indoor localization method based on a convolutional neural network”, *MDPI Sensors*, 2019.
 - [256] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, “Explainable AI: A brief survey on history, research areas, approaches and challenges”, *Springer conference on natural language processing and Chinese computing (CCF)*, 2019.
 - [257] M. Kamal, M. Golam Rabiul Alam, M. Rafiul Hassan, T. Sakib Apon, and M. Mehedi Hassan, “Explainable indoor localization of BLE devices through RSSI using recursive continuous wavelet transformation and XGBoost classifier”, *Future Generation Computer*, 2023.
 - [258] A. Singampalli, D. Gufran, and S. Pasricha, “CIELO: Class-Incremental Continual Learning for Overcoming Catastrophic Forgetting with Smartphone-based Indoor Localization”, *IEEE Access*, 2025.

- [259] A. Singampalli, D. Gufran, and S. Pasricha, “SAFELOC: Overcoming Data Poisoning Attacks in Heterogeneous Federated Machine Learning for Indoor Localization”, *IEEE/ACM Conference on Design, Automation and Test in Europe (DATE)*, 2025.
- [260] A. Singampalli, D. Gufran, and S. Pasricha, “DAILOC: Domain-Incremental Learning for Indoor Localization using Smartphones”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2025.
- [261] D. Gufran, and S. Pasricha, “GATE: Graph Attention Neural Networks with Real-Time Edge Construction for Robust Indoor Localization using Mobile Embedded Devices”, *ACM Transactions on Embedded Computing Systems (TECS)*, 2025.
- [262] D. Gufran, and S. Pasricha, “Towards Explainable Indoor Localization: Interpreting Neural Network Learning on Wi-Fi Fingerprints Using Logic Gates”, *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2025.