Paying Attention to Wildfire: Using U-Net with Attention Blocks on Multimodal Data for Next Day Prediction

Jack Fitzgerald
jack.fitzgerald@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

Ethan Seefried
eseefrie@rams.colostate.edu
Colorado State University
Fort Collins, Colorado, USA

James Yost
jeyost@rams.colostate.edu
Colorado State University
Fort Collins, Colorado, USA

Sangmi Pallickara
sangmi.pallickara@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

Nathaniel Blanchard
nathaniel.blanchard@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

ABSTRACT

Predicting where wildfires will spread provides invaluable information to firefighters and scientists, which can save lives and homes. However, doing so requires a large amount of multimodal data e.g., accurate weather predictions, real-time satellite data, and environmental descriptors. In this work, we utilize 12 distinct features from multiple modalities in order to predict where wildfires will spread over the next 24 hours. We created a custom U-Net architecture designed to train as efficiently as possible, while still maximizing accuracy, to facilitate quickly deploying the model when a wildfire is detected. Our custom architecture demonstrates state-of-the-art performance and trains an order of magnitude more quickly than prior work, while using fewer computational resources. We further evaluated our architecture with an ablation study to identify which features were key for prediction and which provided negligible impact on performance. All of our source code is available on GitHub[1].

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Computer vision**; • **Computer systems organization** → *Distributed architectures.*

KEYWORDS

Wildfire Prediction, Multimodal, Deep Learning Architectures, Attention

ACM Reference Format:
Jack Fitzgerald, Ethan Seefried, James Yost, Sangmi Pallickara, and Nathaniel Blanchard. 2023. Paying Attention to Wildfire: Using U-Net with Attention Blocks on Multimodal Data for Next Day Prediction. In *INTERNATIONAL CONFERENCE ON MULTIMODAL INTERACTION (ICMI '23), October 09–13, 2023, Paris, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3577190.3614116

[1]https://github.com/jfitzg7/paying-attention-to-wildfire

ICMI '23, October 09–13, 2023, Paris, France

ACM ISBN 979-8-4007-0055-2/23/10.
https://doi.org/10.1145/3577190.3614116

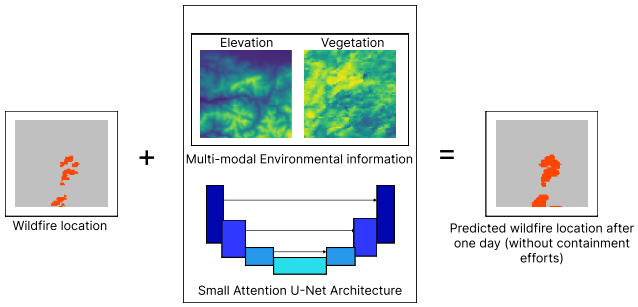

Figure 1: Minimizing the spread of wildfires requires appropriate use of resources to locations the wildfire is likely to spread. In this work, we employ a custom U-net architecture to capitalize on 12 features from multiple distinct modalities in order to predict where a wildfire will spread in the next 24 hours. Modalities include wind and weather predictions alongside area information like vegetation and elevation. Here, we visualize two features (elevation and vegetation) but our model utilizes the full list of all 12 features, shown in Figure 2.

1 INTRODUCTION

The amount of area that has been burned by wildfires has quadrupled in the last four decades [7]. In addition to the threats to human life, the impact of wildfires on the environment is significant; it is estimated that 112 million metric tons of CO_2 is produced due to wildfires in California every year. Experts predict where wildfires will spread so they can deploy measures to prevent said spread. Traditional techniques for tracking the spread of wildfires, such as satellite observation or drone usage, are often slow and prone to noise from wildfire smoke — further, they only facilitate the observation of a fire in real-time, requiring experts to pour over fast swaths of multimodal data in order to predict where the fire will spread next. Modalities that experts consider (beyond simply the wildfire's current location) include wind, elevation, and vegetation, to name a few (examples from the dataset are shown in Figure 2) [11]. In this work, we train a wildfire prediction module to capitalize on this vast amount of data in order to predict where a wildfire will spread over the course of the next 24 hours, as depicted in Figure 1.

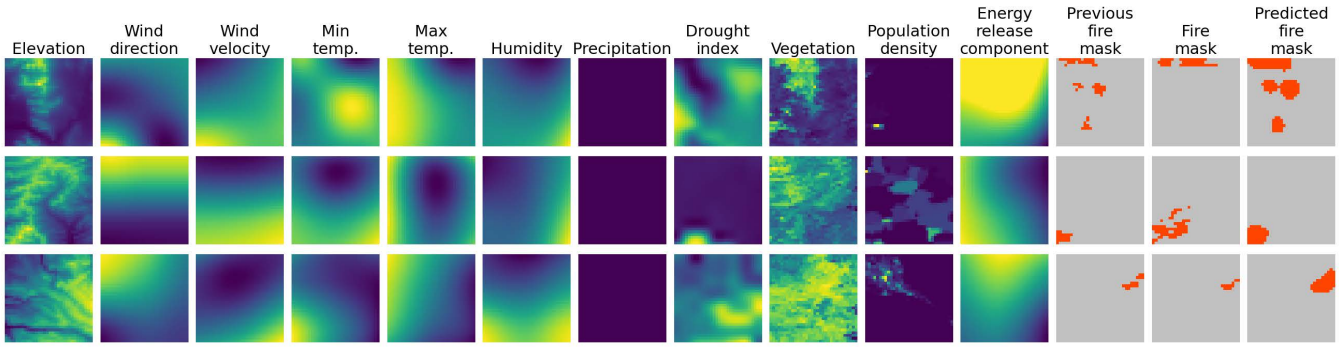

Figure 2: Example images from the data set presented in [15], showing each of the 12 features we derived from our various modalities, and an example prediction.

To address this, we propose a deep learning model that predicts how a wildfire will spread up to one day in advance. The development of a deep learning model for wildfire prediction has significant implications for wildfire management and control. Unlike human experts, an optimized neural network can easily and efficiently parse this multitude of complex features in order to accurately predict how wildfire may spread. In this work, we train a network that is capable of predicting the spread of a wildfire with relative accuracy using minimal data, training time, and compute resources. We combine data from 12 distinct sources [15] (shown in Figure 2) which include a variety of coarse- and fine-grained features from modalities such as vegetation, elevation, and predictions of wind speed, and temperature, among others. By using these features, our model is able to capitalize on the broad range of factors that contribute to wildfire behavior.

The model itself is a custom U-net architecture outfitted with attention heads and a minimal decoding/encoding layers (specifically, 3) — this architecture non-linearly reduces the dimensionality of the full set of features in order to identify key components that facilitate accurate predictions of where the wildfire will spread in the next 24 hours. In comparison with prior work, we achieve state-of-the-art performance predicting where wildfire will spread and we are able to train our model far more efficiently [15].

In particular, the short training time is critical to train a set of new models for regions with high geospatial variability. Prior work required nearly 1000 epochs of training [15], with the addition of augmented data which has its own generation cost, while our architecture needs only 20 epochs to train, achieving similar performance. The fast training time makes it easy to train a model quickly, which is essential for cases where the model needs to be fine-tuned to adapt to different domains — for example, the characteristics of how a wildfire will spread in The Great Smoky Mountains (situated in the east of North America) is likely to be distinct from how it will spread in California (which is on the west coast of North America), where our current data set was collected.

We also investigate the importance of each modality for predicting wildfire spread, finding that performance is relatively stable even with as few as three modalities. The identification of key features associated with wildfire spread has the potential to improve future models even further. Based on our findings, we provide several recommendations for future work.

Our contributions can be summarized as follows:

- We designed a novel architecture for predicting how a wildfire will spread over a 24-hour period. Our model achieves state-of-the-art results for predicting the spread of wildfires.
- Our network achieves an order of magnitude decrease in training time over prior state-of-the-art work (20 epochs compared with 1000 epochs [15]). We were able to achieve this training speedup with far inferior hardware: we trained with 6 GTX 1060s, while [15] trained with 4 V100s.
- We investigate how variants of our architecture improve wildfire spread prediction performance.
- We ablate the number of features and find that using fewer modalities can yield comparable results, indicating our model should be robust when fewer data sources are available. Further, this analysis provides key information about which features are the most important for predicting the spread of wildfires.
- We anticipate our findings and analyses will inform the development of future wildfire prediction systems, as well as measures for limiting the spread of wildfires.

2 RELATED WORK

For quite a time, humankind has strived to forecast the behavior of fires to mitigate their destructive effects. Early models from the 1990s employed mathematical and probabilistic predictors to estimate the spread of fire [8]. Among these, the "Fire Area Simulator-model development and evaluation" (FARSITE) has gained prominence and has been leveraged by the United States Department of Agriculture (USDA) [11]. In recent years, more sophisticated methods such as machine learning, deep learning, and genetic algorithms have been employed to model wildfire propagation [19, 20].

However, machine learning models necessitate a significant amount of data to attain optimal performance, particularly when applied to wildfires [22]. This implies the need for vast and diverse data sets from various sources such as topography, vegetation, weather data, drought index, and population density, among others [14, 15]. The rising trend of using deep learning techniques for fire

prediction is accompanied by an increase in costs, both in terms of time and energy, due to the need for extensive training [25]. Hence, it is crucial to have models that can be trained quickly and inexpensively for ease of use.

Other works have taken the approach of numerical simulation to predict wildfire spread, however, this approach is too computationally complex to train, making it difficult to adapt to a new environment in real time [4]. In contrast, [15] takes advantage of the vast amount of remote-sensing data that is currently available to create a real world data set to predict wildfire spread. Other approaches that have worked using the data set in [15] have looked to expand upon it, rather than using it for predictions. In [3], they introduce new modalities, and also improve the spatial resolution of the fire masks. However, they did not test any models on their new data set. Similarly, other approaches have been limited in their prediction field, predicting a subset of total fires, such as only bushfires [24].

An interesting alternative approach to detecting wildfires is described in [16], which proposes a network of low-powered IoT devices that can detect smoke and sudden increases in temperature in order to identify wildfires. The devices can also measure humidity, carbon dioxide, light, precipitation, and wind speed data. Setting up and maintaining such a network of devices at a large scale could prove costly, but it would provide a more fine-grained temporal resolution by collecting data every 15 minutes, for example. The devices are also able to collect much of the data that is related to wildfire spreading without the need to compile information from various sources and satellites.

To address these challenges, we propose a novel approach to the problem of wildfire prediction, working with the data set presented in [15]. Our approach utilizes a custom variant of the U-net architecture, which we call the Wildfire Prediction Network (WPN), to limit the amount of data required while still achieving similar performance levels.

3 DATA SET

The data set titled "Next Day Wildfire Spread" [15] contains 12 input features and one target feature, all of which are derived from several modalities. The features themselves are essentially 64 x 64 dimensional images, where each pixel represents a 1km x 1km area. The modalities themselves come from several sources and can be identified as topography/elevation, weather, drought indices, vegetation indices, population density, energy release component (ERC), and fire masks. The weather modality covers several features including wind speed, wind direction, minimum temperature, maximum temperature, humidity, and precipitation. The rest of the modalities are themselves input features. Figure 2 shows 3 samples that contain all of the input features: elevation, wind direction, wind speed, minimum temperature, maximum temperature, humidity, precipitation, drought index, vegetation, population density, energy release component, and the previous fire mask. The target feature, fire mask, is the next day fire mask that is predicted by our models.

The entire data set was retrieved using Google Earth Engine (GEE) by aggregating data from various sources. It is only 3GBs in size, and this includes the training, evaluation, and testing data sets. All of the data is collected from areas within the contiguous United States from 2012 to 2020. The historical wildfire data comes from the MOD14A1 V6 data set [13], which provides daily fire masks since 2000 with a spatial resolution of 1km. In the fire masks, values of -1 represent missing data, 0 represents no fire, and 1 represents fire. Topography data comes from the Shuttle Radar Topography Mission (SRTM) [10] and provides elevation data at a spatial resolution of 30 meters. Weather data is obtained from the GRIDMET data set [1], which provides daily records since 1979 for temperature, precipitation, wind, and humidity at a spatial resolution of 4km. Drought data is obtained from the GRIDMET Drought data set [2] at a spatial resolution of 4km and is collected every five days since 1979. The drought indices are derived from the GRIDMET data set itself. Vegetation data is from the VNP13A1 data set [9] which provides vegetation indices collected every eight days since 2012 with a spatial resolution of 500 meters. Population density data was obtained from the Gridded Population of World Version 4 (GPWv4) data set [12], which is collected every five years at a spatial resolution of 1km. Finally, the energy release component (ERC) data is obtained from the National Fire Danger Rating System (NFDRS) [6]. All of the features were adjusted to match the spatial and temporal resolutions of the fire masks. [15] has more details about the various sources of the data and the pre-processing techniques used to adjust all of the features.

4 METHODOLOGY

Task Definition. The task of wildfire prediction aims to predict the next day's fire mask, given multimodal input. The input consists of 12 inputs derived from seven modalities: Fire masks, topography (elevation), Weather (temps, wind, precipitation, humidity), drought indices, vegetation indices, and population density. Each input can be denoted as a matrix: $I_m \in \mathbb{R}^{32x32}$ where m represents the feature.

4.1 Overall Description

The overview of our WPN model is shown in Figure 3. It is essentially a U-Net style architecture that consists of two main components, the encoder and decoder. The main building block of the encoder and the decoder is called the double-convolution (double-conv) block. The double-convolution block consists of two convolutional layers that use a kernel size of 3, a stride of 1, and a padding of 1. The kernel size, stride, and padding are chosen such that the dimensions of the input are not altered. Both of the convolutional layers in the double-convolution block are followed by a batch normalization layer and a ReLU activation function. In the encoder portion, the double-convolution blocks are followed by max-pooling layers with a kernel size of 2 and a stride of 2. In the decoder portion, the double-convolution blocks are followed by an upsampling block.

A key component of our model is the addition of attention blocks. Before the skip connections are passed to the corresponding decoder blocks, they are passed into an attention block and combined with the upsampled output from the previous double-convolution block. The main idea behind the attention blocks is that they help the model learn what low-level features from the encoding layers to pay attention to, and the intention is that this will improve the accuracy of the model without too much additional computational overhead.

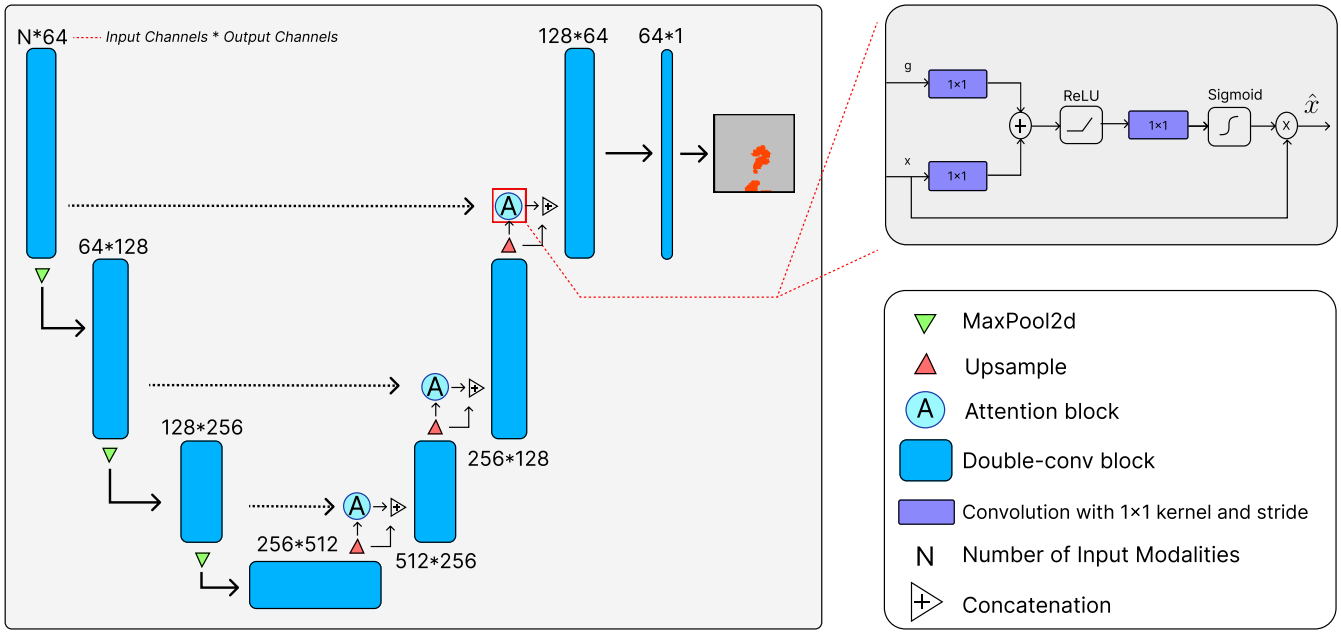

Figure 3: Wildfire Prediction Network (WPN) architecture.

4.2 The Encoder

The encoder portion of the model consists of three double-convolution blocks, each of which are followed by max pooling operations. The first double-convolution block receives a certain number of inputs from the modalities, and it treats each I_m input as a single channel in a 32x32 dimensional image. It then passes the input through the double-convolution block and creates 64 new output channels before passing it to the first max pooling layer. The output of the first max pooling layer is passed to the second double-convolution block, which receives as input a 16x16 image with 64 channels. The second double-convolution block doubles the number of output channels to 256, and passes its output to the second max pooling layer, further reducing the image size. Finally, the third double-convolution block receives an 8x8 image with 128 channels, and it again doubles the number of output channels to 256 and passes its output to the final max pooling layer. The output of the final max pooling layer is passed to the base block, another double-convolution block that receives a 4x4 image with 256 channels as input.

4.3 The Decoder

Before the decoder portion begins, the base block processes the final output from the encoder, and it again doubles the number of output channels to 512. The output of the base block is then passed into an upsampling block before being handed off to the first attention block and double-convolution block in the decoder. Apart from the double-convolution blocks, the primary components of the decoder are the upsampling blocks, the skip connections, and the attention blocks.

4.3.1 Upsampling Blocks. The base block and each double-convolution block in the decoder utilize the upsampling block. The upsampling block consists of an upsample operation with a scale factor of 2, followed by a convolutional layer with a kernel size of 3, a stride of 1, and a padding of 1. The convolutional layer's output channels are half the number of input channels. The output of the convolutional layer is passed through a batch normalization layer and finally through the ReLU activation function. The upsampling block's effect is that it increases each dimension of the input image by a factor of 2 and halves the number of channels. For example, a 4x4 image with 512 channels will be converted into an 8x8 image with 256 channels.

4.3.2 Skip Connections. The output from each of the double-convolution blocks in the encoder is passed through an attention block before being concatenated with the output of a corresponding upsampling block. These concatenations are referred to as skip connections. The output from the third double-convolution block in the encoder will be concatenated with the output of the first upsampling block, the second is concatenated with the second upsampling block, and the first is concatenated with third upsampling block. In essence, the skip connections are using the low-level features from the encoder to help the decoder recover spatial information that might have been lost during the downsampling process.

4.4 The Attention Blocks

Before the output from the encoder layers are passed to the decoder layers via the skip connections, they are passed through attention blocks. As can be seen in Figure 3, the two inputs to the attention block are g and x. The input g is the output from the previous upsampling block, and the input x is coming from the corresponding double-convolution block in the encoder. Both g and x will have the same dimensions and number of channels, and they are passed into separate convolutional layers with a kernel size of 1, a stride of

1, and a padding of 0. These convolutional layers' output channels are half the number of input channels. The output from both of these convolutional layers are then added together element-wise, and then passed through the ReLU activation function. The output of the ReLU is then passed into another convolutional layer with a kernel size of 1, a stride of 1, and a padding of 0, and the number of output channels is 1. The output of the final convolutional layer is passed through a batch normalization layer and is then passed through a sigmoid activation function. The output of the sigmoid is then multiplied element-wise with the original input x to create the final output $\hat{x}$. Finally, $\hat{x}$ is concatenated with g, completing the skip connection and passing on to the corresponding double-convolution block. The main idea is that the attention blocks will learn what areas of the input x to focus on by assigning larger weights to those areas.

4.5 Model Training

There was a total of 12 different modalities (I_m) in the data set, much of which we thought was redundant information. We experimented with reducing the number of modalities to see how this would impact the overall performance of the models. The performance of a model was measured by the precision, recall, and F1 score for the fire class, and by the mean and max IOU scores. The main metric that we focused on was the F1 score. All of the experiments used: focal loss with an alpha value of 0.85 and a gamma value of 2, *RMSprop* with a learning rate of 0.001 and momentum of 0.9, and trained for 20 epochs using a batch size of 50. The model was trained on all possible 90-degree rotations for each input sample (32x32 pixels) in the training data set.

4.5.1 Data Augmentation. The original data set had 64 x 64 images, but we found strange behavior in the model with this set; see Appendix A. Because of this, we took random crops of size 32 x 32 of the original samples. To perform the crops, we would first calculate random offsets for each sample and store them in a map where the key was the sample index, and the values were the random x and y offsets. At runtime, the random crops would be applied to the samples and then passed to the model to then be trained on. Finally, we would check to ensure that the random crops did not have any missing data in the target fire masks; any samples with missing data would be discarded. This approach contrasts the approach that [15] took; they simply ignored any pixels that had a value of −1 (the value -1 represents missing data) in the target fire mask when performing the cross-entropy loss calculation. We chose our approach for simplicity and ease of training. To increase the amount of data and make the model more robust, we rotated each sample by 0, 90, 180, and 270 degrees.

4.5.2 Distributed Training. All of the models were trained using PyTorch's Distributed Data Parallel (DDP) API. First, we had to set environment variables that specified the IP address and port of the master node. Second, we had to specify that we wanted to use the NVIDIA Collective Communication Library (NCCL) backend when initializing the processes. We used machines that have GTX 1060s with 6GB of VRAM to train the models. Typically, anywhere between 4 − 8 of these machines were used to train our models in parallel.

5 EXPERIMENTS

5.1 Imbalanced Classification

Our exploratory analysis of the dataset revealed that 97% of all pixels in the target fire mask are the "no fire" class, while the remaining 3% are the "fire" class. This indicates an imbalanced classification problem, making it more challenging to develop and train models to deal with the imbalance. However, there are ways to handle imbalanced classification, such as oversampling and special loss functions.

5.1.1 Oversampling and Undersampling. Oversampling and undersampling are two ways that can be used to deal with imbalanced classification. However, the fact that the target fire mask is a 2D image makes it slightly more difficult to choose which samples should be oversampled or undersampled. One way to approach oversampling for this problem is to calculate the percentage of fire pixels in a particular target fire mask, and only select samples that exceed a certain threshold, say 50%. However, we found that there are typically only a handful of samples that have target fire masks with greater than 50% fire pixels, so selecting a lower threshold is necessary to achieve a larger pool of samples to choose from. We only used oversampling during early testing and exploration, but decided to mention it here due to the interesting difficulties we encountered.

5.1.2 Focal Loss. An approach to dealing with imbalanced classification is to use a special loss function, such as weighted cross-entropy so that misclassifying a fire pixel will result in a higher loss. Doing this causes the model to want to predict fire more often, since false positives are not as punishing as false negatives. Another function that is similar to cross-entropy is focal loss [17]. Focal loss lets the model focus on hard examples that it has a difficult time trying to classify, i.e. pixels that have a low probability of being guessed correctly. Due to this aspect of focal loss, we decided to use it instead of weighted cross-entropy. The equation for focal loss can be defined as:

$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t) \tag{1}$$

If we let $p \in [0,1]$ represent the probability of the class with $y = 1$ (The fire class), then p_t can be defined as:

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ (1-p), & \text{otherwise} \end{cases} \tag{2}$$

We can also use a weighting factor $\alpha \in [0,1]$ that represents the weight for class 1 (the fire class) and $(1-\alpha)$ for class 0 (the no fire class). We can then let α_t be defined similarly to p_t:

$$\alpha_t = \begin{cases} \alpha, & \text{if } y = 1 \\ (1-\alpha), & \text{otherwise} \end{cases} \tag{3}$$

Using the weighting factor α_t, the focal loss can then be redefined as:

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{4}$$

The two hyperparameters in focal loss that need to be tuned are α and γ. When $\gamma >= 1$, it controls the probability threshold that defines what a well-classified example is. The higher the value of γ, the lower the probability needs to be for a prediction to be

considered well-classified. We found that setting γ to 2 and keeping α somewhere between 0.8 and 0.95 gave accurate predictions.

5.2 Baseline & Metrics

The primary models that we chose to compare our model, Wildfire Prediction Network (WPN) with used the U-Net architectures from [5, 18, 21].

- **U-Net:** A standard U-Net architecture, acting as the baseline model.
- **R2U-Net:** Baseline U-Net, with only recurrent residual blocks added to the encoder and decoder.
- **AttU-Net:** Baseline U-Net, with only attention blocks added to the decoder.
- **R2AttU-Net:** Baseline U-Net with attention blocks added to the decoder, and recurrent residual blocks added to the encoder and decoder.

We used the F1 score as our primary metric to evaluate each of the models. As we have an imbalanced classification problem, the F1 score proves to be an effective way of evaluating the model's accuracy. For the task of wildfire prediction, we need to take into account both the precision and recall of the model and the F1 provides a balanced metric to evaluate them both.

6 RESULTS

6.1 All Features

Using all 12 input features resulted in accurate predictions, and the best model evaluated on the F1 score was WPN with a value of 0.361. Table 1 shows the performance of the models that were trained on all of the 12 input features. R2U-Net had the best recall of 48.8%, and WPN had the best precision of 30.3%.

6.2 7 Features

Reducing the number of input features from 12, down to 7 (elevation, wind speed, humidity, drought, vegetation, pop. density, and prev. fire mask) degraded the precision and improved the recall of all the models, except for R2AttU-Net and WPN which had better precision and worse recall. Overall, the F1 scores went down compared to the models trained on all 12 features. R2AttU-Net had the best F1 score of 0.346 with a precision of 29.7% and recall of 41.6%. R2U-Net had worse recall than the standard U-Net, and this was the only experiment where that was the case.

6.3 3 Features

The results of using only the elevation, vegetation, and previous fire mask features can be seen in Table 1. Unexpectedly, nearly all of the models achieved their highest F1 scores using only these 3 features (ignoring the results from Appendix A). However, the R2AttU-Net model actually received its worst F1 score in this experiment, which is surprising considering the other models performed quite well. The standard U-Net achieved its best precision score of 29.7%, but its worst recall score of 46.9%. The same was true for AttU-Net which received a precision of 30.9%, and a recall of 46%. Both AttU-Net and WPN received the overall best F1 score of 0.370, with WPN having a slightly better precision score and a slightly worse recall score than AttU-Net. Although WPN and AttU-Net achieved the same F1-score in this experiment, the training time and F1 score demonstrate the capabilities of WPN to predict wildfires given a small number of inputs.

Figure 4: Prediction with only Prev. Fire Mask

7 FURTHER ANALYSIS

7.1 Previous Fire Mask

One of the features that the models have as input is the previous fire mask, which was the most important piece of information when the model was making a prediction. We tested this by training the model only on the previous fire mask, and all features besides the previous fire mask .

We can see in Table 1 that the overall performance of the models are relatively good with just the previous fire mask. Our WPN model achieved an F1 score of 0.361, which is around average for most of our models. We can see the strategies that the models learned in Figure 4. If there was a fire, it generally just predicts that the fire got bigger.

7.2 Focal Loss vs. Weighted Binary Cross Entropy

An unexpected benefit that focal loss had over-weighted BCE was that it drastically reduced the training time. We found that using a γ value of 2 for the focal loss likely reduced the loss of many of the predictions to 0, and PyTorch must've been able to take advantage of that by reducing the number of weights that needed to be updated during back-propagation. To test this theory we also tested the focal loss with a γ of 0, which is conceptually the same as

Models (32x32)	Total Params.	Modalities	Precision	Recall	F1 Score	Mean IOU	Max IOU
U-Net	34.5M	12	28.2%	47.8%	0.355	0.1499	1.0
		7	21.9%	54.2%	0.311	0.1479	0.7894
		3	29.7%	46.9%	0.364	0.1447	0.7857
		Prev. fire mask	29.6%	46.3%	0.361	0.1575	0.75
R2U-Net	39.1M	12	25.9%	48.8%	0.338	0.1459	1.0
		7	22.8%	52.6%	0.328	0.1494	1.0
		3	25.5%	51.1%	0.341	0.1420	1.0
		Prev. fire mask	35.3%	34.4%	0.349	0.1354	0.82
AttU-Net	34.9M	12	28.2%	48.3%	0.356	0.1481	0.8
		7	23.9%	50.9%	0.326	0.1508	0.75
		3	30.9%	46.0%	**0.370**	0.1455	0.7404
		Prev. fire mask	29.4%	46.7%	0.361	0.1574	0.86
R2AttU-Net	39.4M	12	28.3%	47.6%	0.355	0.1431	1.0
		7	29.7%	41.6%	**0.346**	0.1489	1.0
		3	24.2%	54.6%	0.335	0.1456	0.8571
		Prev. fire mask	32.2%	40.2%	0.358	0.1532	1.0
WPN	8.7M	12	30.3%	44.8%	**0.361**	0.1444	0.8947
		7	31.8%	35.9%	0.337	0.146	0.7
		3	31.4%	45.1%	**0.370**	0.1489	0.7143
		Prev. fire mask	28.8%	48.1%	0.361	0.1563	0.81

Table 1: Results for the models trained on the 32x32 data set, evaluated using the F1 score and the mean and max IOU metrics.

Loss function	Training time
Focal loss[WPN] (gamma = 2)	1.75 minutes per epoch
Focal loss (gamma = 2)	10 minutes per epoch
Weighted BCE (weight = 5)	1 hour per epoch
Focal loss (gamma = 0)	1 hour per epoch

Table 2: Differences in training time when using 4-6 nodes with 1 GTX 1060 each with focal loss and weighted binary cross entropy

using binary cross-entropy, and found that this also increased the training time dramatically. This seems to confirm the theory that fewer weights need to be updated when the γ parameter is greater than 0. This fact about focal loss makes it a huge improvement over weighted binary cross entropy. Table 2 shows the differences in training times between the two loss functions. When using 4 nodes with a single GTX 1060 on each, It took nearly an hour per epoch to train the standard U-Net model using weighted BCE and focal loss with a gamma value of 0, while only taking 10 minutes per epoch when using focal loss with a gamma value of 2. Our Wildfire Prediction Network (WPN) model achieved significant improvement in training times when using focal loss with a γ value of 2, taking on average 1.75 minutes using between 4 - 6 nodes.

8 EVALUATION

8.1 Feature Selection

It seems clear from the experiments that some features must be combined to allow the models to make the best predictions. However, discovering what this combination is through brute force can be a time-consuming process since there are $2^{12} - 1 = 4095$ possible combinations. Perhaps only a handful of the features will give the best performance. Testing all combinations could be rewarding just to discover the best one, but it is doubtful that the performance would improve significantly. The more interesting question to ask would then be, why does this combination of features give the best performance?

Judging from the feature ablation experiments in [15], it does appear that the elevation and vegetation features work well when used with only the previous fire mask. This is primarily why we chose to use them together in our experiment in Section 6.3, and our results do appear to confirm their findings. Our best-performing models, in terms of the F1 score, were trained with only elevation, vegetation, and the previous fire mask. Performing further analysis to discover why this is the case could potentially reveal interesting patterns that could prove useful.

8.2 Insights On the Model Variants

8.2.1 Wildfire Prediction Network (WPN) further speeds up training and achieved the best F1 score. Removing blocks from the encoder and decoder of the AttU-Net model seemed to positively affect the F1 scores. While the recall scores for WPN were not as high as the recall scores for AttU-Net, its precision scores were slightly better and more consistent in every experiment. The training times were also significantly faster when using WPN. It took 1.75 minutes per epoch to train WPN vs. 8 minutes per epoch to train AttU-Net when using four nodes each having 1 GTX 1060.

8.2.2 The recurrent residual models achieved higher maximum IOU scores. Most of the recurrent residual (R2) models were able to achieve perfect IOU scores on at least some of the examples, which

is in contrast to the other models where only the 12-feature U-Net model was able to achieve a perfect IOU score. It's not clear why the R2 models exhibited this behavior, but it could mean that there are certain examples that they have an easier time dealing with. However, It's also interesting that the mean IOU scores for the R2 models are generally slightly worse when compared to the other models. The relatively low mean IOU scores appear to be an indication that the R2 IOU scores aren't better overall.

8.2.3 The synergy between the recurrent residual and attention blocks is unclear. In all of the experiments, the R2AttU-Net variant either had a better recall or better precision over both R2U-Net and AttU-Net. In addition, either the recall or the precision was worse than both R2U-Net and AttU-Net. This behavior seems to suggest that the recurrent residual blocks and the attention blocks are not particularly synergistic for this data set. However, this could be due to the fact that improving precision typically reduces recall and vice versa. Guessing fire less but more precisely can lead to more false negatives, and guessing fire more often typically leads to more false positives.

8.2.4 The previous fire mask only models have slightly better mean IOU scores. The models that were trained on only the previous fire mask had slightly higher mean IOU scores than their counterparts (except for in the R2U-Net models). As can be seen in both Figure 2 and Figure 4, the models typically predict that the fire will grow around the original regions in the previous fire mask. Perhaps the previous fire mask only models tend to keep predictions tighter to the original regions, while models trained with the other features are more inclined to make more spread out predictions.

9 DISCUSSION

One of the limitations of the data set itself is the spatial and temporal resolutions. Fire maps from x number of previous days would be needed before a time series model could start making predictions, which means that the models might not be useful until the fire has been raging for several days. Moreover, The spatial resolution of 1km may be too coarse for the models to make accurate assessments of the fires. In [3], they improved upon several facets of the data set from [15]. They reduced the spatial resolution of the fire masks from 1km down to 375m by utilizing VIIRS S-NPP [23] instead of the MOD14A1 v6 data set, and they were also able to introduce new modalities such as short wave radiation and burn index. In future work, we will seek to utilize this new data set to compare the differences in performance between models trained on the data set from [15] versus the data set from [3].

Another problem that was noted by [15] is the fact that many of the fires are likely being actively fought. There are no indications in the data set of where active firefighting efforts are occurring, so it's likely that the models will not learn how wildfires spread naturally. A potential solution to this problem is somehow recording where firefighting efforts are taking place, but we are unaware of any current data sets that track this information. Moreover, this hypothetical data set would need to be compatible with the spatial and temporal resolutions of the fire masks.

Prior work achieved strong performance but at a large time and resource cost during training. Specifically, [15] was trained for 1000 epochs and 1000 iterations per epoch with 4 V100 GPUs. One of our major contributions is the increased efficiency of training — the models we tested obtained similar results after training for only 10 - 20 epochs, and the overall training time took roughly 10 minutes per epoch (2 minutes per epoch for our Wildfire Prediction Network) when using 4 GTX 1060s. The usage of focal loss, in combination with the small network size of our Wildfire Prediction Network (WPN) model, drastically improved training times. In particular, we believe that focal loss is essential to minimizing training time and we recommend future work utilize it.

Future works should explore using time series models with larger sequences of input data. We were unable to explore this because the data set we used only consisted of one day's worth of data. Additionally, it may be possible to further reduce training time by decreasing the model size (we obtained a reasonable speedup moving from four to three layers in the WPN encoder/decoder). Wildfire prediction in the real world would benefit from lower computing costs, as running a model on something such as a smartphone may allow firefighters to have easy access to real-time predictions. Ideally, this will lead to faster response times, guiding firefighters to the predicted spread to slow or stop the fire altogether.

10 CONCLUSION

Wildfires are a major global threat and accurate predictions of where they will spread is critical. However, traditional methods of wildfire prediction are often unreliable due to the complexity and unpredictability of wildfire behavior and the need for large amounts of time and computational resources. This paper proposes a deep-learning architecture for the purpose of predicting wildfire spread over the course of a day with relatively minimal time and computing resources needed. To demonstrate the capabilities of our approach, we used a multimodal data set curated by [15]. We find our architecture trains significantly faster than prior work while also achieving state-of-the-art performance. Our study represents an important step forward in developing machine learning-based approaches to wildfire prediction, and we hope it will inspire further research in this area.

ACKNOWLEDGMENTS

This research was supported by the DARPA Geometries of Learning Program [HR00112290074], the National Science Foundation [OAC-1931363, CNS-2312319, 2023-67021-39829], and the National Institute of Food and Agriculture [COL0-FACT-2019].

REFERENCES

[1] John T Abatzoglou. 2013. Development of gridded surface meteorological data for ecological applications and modelling. *International Journal of Climatology* 33, 1 (2013), 121–131. https://doi.org/10.1002/joc.3413

[2] John T Abatzoglou, David E Rupp, and Philip W Mote. 2014. Seasonal climate variability and change in the Pacific Northwest of the United States. *Journal of Climate* 27, 5 (2014), 2125–2142. https://doi.org/10.1175/JCLI-D-13-00218.1

[3] Syed Haider Ali, Anish Goel, Aditya Singirikonda, Ali Y Khan, and Ting Xiao. 2022. Towards a Comprehensive Dataset for Next-Day Wildfire Prediction. In *2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*. IEEE, 593–598. https://doi.org/10.1109/QRS-C57518.2022.00095

[4] Frédéric Allaire, Vivien Mallet, and Jean-Baptiste Filippi. 2021. Emulation of wildland fire spread simulation using deep learning. *Neural networks* 141 (2021), 184–198. https://doi.org/10.1016/j.neunet.2021.04.006

[5] Md Zahangir Alom, Mahmudul Hasan, Chris Yakopcic, Tarek M Taha, and Vijayan K Asari. 2018. Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation. *arXiv preprint arXiv:1802.06955* (2018). https://doi.org/10.48550/arXiv.1802.06955

[6] Larry S Bradshaw, John E Deeming, Robert E Burgan, D Jack, et al. 1984. The 1978 national fire-danger rating system: technical documentation. *General Technical Report INT-169. Ogden, UT: US Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station. 44 p.* 169 (1984). https://doi.org/10.2737/int-gtr-169

[7] Marshall Burke, Anne Driscoll, Sam Heft-Neal, Jiani Xue, Jennifer Burney, and Michael Wara. 2021. The changing risk and burden of wildfire in the United States. *Proceedings of the National Academy of Sciences* 118, 2 (2021), e2011048118. https://doi.org/10.1073/pnas.2011048118

[8] NP Cheney, JS Gould, and Wr R Catchpole. 1998. Prediction of fire spread in grasslands. *International Journal of Wildland Fire* 8, 1 (1998), 1–13. https://doi.org/10.1071/WF9980001

[9] A Didan and A Barreto. 2018. VIIRS/NPP Vegetation Indices 16-Day L3 Global 500m SIN Grid V001. *NASA EOSDIS Land Processes DAAC: Oak Ridge, TN, USA* (2018). https://doi.org/10.5067/VIIRS/VNP13A1.001

[10] Tom G Farr, Paul A Rosen, Edward Caro, Robert Crippen, Riley Duren, Scott Hensley, Michael Kobrick, Mimi Paller, Ernesto Rodriguez, Ladislav Roth, et al. 2007. The shuttle radar topography mission. *Reviews of geophysics* 45, 2 (2007). https://doi.org/10.1029/2005RG000183

[11] Mark A Finney. 1998. *FARSITE, Fire Area Simulator–model development and evaluation.* Technical Report 4. https://doi.org/10.2737/rmrs-rp-4

[12] Center for International Earth Science Information Network CIESIN Columbia University. 2016. Gridded Population of the World, Version 4 (GPWv4): Population Density, Revision 11. (2016). https://doi.org/10.7927/H4NP22DQ

[13] Louis Giglio and Christopher Justice. 2015. MOD14A1 MODIS/Terra thermal anomalies/fire daily L3 global 1km SIN grid V006. *NASA EOSDIS Land Processes DAAC* 10 (2015).

[14] Jonathan L Hodges and Brian Y Lattimer. 2019. Wildland fire spread modeling using convolutional neural networks. *Fire technology* 55 (2019), 2115–2142. https://doi.org/10.1007/s10694-019-00846-4

[15] Fantine Huot, R Lily Hu, Nita Goyal, Tharun Sankar, Matthias Ihme, and Yi-Fan Chen. 2022. Next day wildfire spread: A machine learning dataset to predict wildfire spreading from remote-sensing data. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), 1–13. https://doi.org/10.1109/TGRS.2022.3192974

[16] Tareq Khan. 2023. Ultra-Low-Power Architecture for the Detection and Notification of Wildfires Using the Internet of Things. *IoT* 4, 1 (2023), 1–26. https://doi.org/10.3390/iot4010001

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. (2017), 2980–2988. https://doi.org/10.48550/arXiv.1708.02002

[18] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. 2018. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999* (2018). https://doi.org/10.48550/arXiv.1804.03999

[19] Jorge Pereira, Jérôme Mendes, Jorge SS Júnior, Carlos Viegas, and João Ruivo Paulo. 2022. A review of genetic algorithm approaches for wildfire spread prediction calibration. *Mathematics* 10, 3 (2022), 300. https://doi.org/10.3390/math10030300

[20] David Radke, Anna Hessler, and Dan Ellsworth. 2019. FireCast: Leveraging Deep Learning to Predict Wildfire Spread.. In *IJCAI*. 4575–4581. https://doi.org/10.24963/ijcai.2019/636

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

[22] Younes Oulad Sayad, Hajar Mousannif, and Hassan Al Moatassime. 2019. Predictive modeling of wildfires: A new dataset and machine learning approach. *Fire safety journal* 104 (2019), 130–146. https://doi.org/10.1016/j.firesaf.2019.01.006

[23] Wilfrid Schroeder, Patricia Oliva, Louis Giglio, and Ivan A Csiszar. 2014. The New VIIRS 375 m active fire detection data product: Algorithm description and initial assessment. *Remote Sensing of Environment* 143 (2014), 85–96. https://doi.org/10.1016/j.rse.2013.12.008

[24] Saroj Kumar Sharma, Jagannath Aryal, Quanxi Shao, and Abbas Rajabifard. 2023. Characterizing Topographic Influences of Bushfire Severity Using Machine Learning Models: A Case Study in a Hilly Terrain of Victoria, Australia. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 16 (2023), 2791–2807. https://doi.org/10.1109/JSTARS.2023.3249643

[25] Tien-Ju Yang, Yu-Hsin Chen, Joel Emer, and Vivienne Sze. 2017. A method to estimate the energy consumption of deep neural networks. In *2017 51st asilomar conference on signals, systems, and computers*. IEEE, 1916–1920. https://doi.org/10.1109/ACSSC.2017.8335698

A RESULTS FROM USING THE 64X64 SAMPLES

The models achieved some of the best F1 scores when trained on the unaltered 64x64 features, which can be seen in Table 3. But despite the models getting some of the best precision, recall, and F1 scores, the 64x64 target fire masks suffer from a lack of variety. Looking at the distribution of where the fires are in the target fire masks

Models (64x64)	Total Params.	Modalities	Precision	Recall	F1 Score	Mean IOU	Max IOU
U-Net	34.5M	12	34.4%	48.0%	**0.401**	0.2311	0.75
		7	33.5%	46.9%	0.391	0.2290	1.0
		3	28.9%	57.9%	0.386	0.2277	0.7059
		Prev. fire mask	33.5%	46.7%	0.390	0.2228	1.0
R2U-Net	39.1M	12	27.2%	57.4%	0.369	0.2033	0.75
		7	26.8%	57.4%	0.365	0.1996	0.6875
		3	31.5%	48.9%	0.383	0.2181	1.0
		Prev. fire mask	27.8%	42.8%	0.337	0.1356	0.8421
AttU-Net	34.9M	12	31.1%	52.8%	0.391	0.2305	0.7727
		7	31.1%	53.2%	0.393	0.2301	0.7778
		3	31.5%	53.4%	0.396	0.2354	0.8333
		Prev. fire mask	27.1%	56.7%	0.367	0.2115	0.7391
R2AttU-Net	39.4M	12	27.1%	57.8%	0.369	0.2072	0.7857
		7	29.7%	55.1%	0.386	0.2245	0.8947
		3	28.7%	54.3%	0.376	0.2119	0.7857
		Prev. fire mask	28.9%	35.7%	0.319	0.1289	1.0
WPN	8.7M	12	33.2%	50.5%	**0.401**	0.2267	0.6316
		7	30.1%	52.7%	0.383	0.1955	1.0
		3	31.2%	52.5%	0.392	0.2281	0.7143
		Prev. fire mask	33.8%	47.8%	0.396	0.2325	0.7143

Table 3: Results for the 64x64 models, evaluated using the F1 score and IOU metric.

and the predictions that the models make, reveals some extreme bias. Most of the fires in the target fire masks occur in the center, so the models learn to predict fire in the center on most samples. It is unclear why the original target fire masks have this distribution of fires. Perhaps it is an error in the way the data was collected from GEE, or in our dataset, the orientations of samples were limited by the geospatial extent in the original data collection. Figure 5 shows the distribution of fires in the unaltered 64x64 target fire masks for all of the data sets, and Figure 6 shows the distribution of fire predictions on the testing data set from a subset of the models. Figure 6 highlights the undesirable strategy that the models learned, which was frequently guessing fire in the middle of the fire mask.

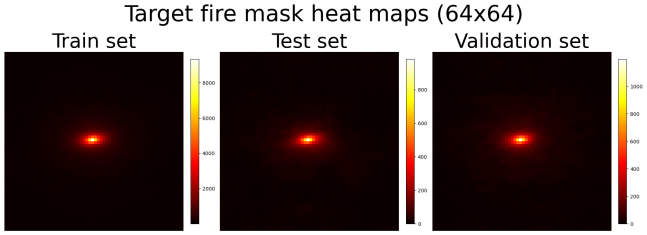

Figure 5: Heat maps showing the distribution of fire in the unaltered 64x64 target fire masks from the training, testing, and validation sets, respectively.

Our baseline experiments used randomly cropped 32x32 areas from the original 64x64 samples. [15] noted that the fires were mostly contained in the center of the original 64x64 target fire masks, so taking random 32x32 crops was used to counteract this problem. Figure 7 shows the distribution of fires in a set of randomly cropped 32x32 target fire masks for all of the data sets. The distribution of fires is much more spread out compared to the original 64x64 target fire masks, which can be seen in 5. This is a much more desirable-looking distribution since the models need to adopt more meaningful strategies other than simply predicting fires in the middle. Figure 8 shows the distribution of fire predictions for the test data set when using a subset of the U-Net variants that were trained on all 12 modalities.

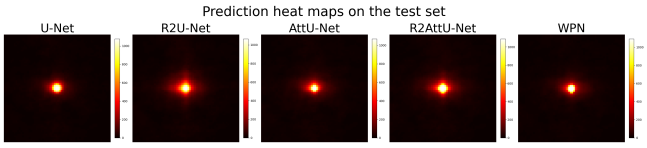

Figure 6: Heat maps showing the distribution of fire predictions for the unaltered 64x64 test set from the U-Net, R2U-Net, AttU-Net, R2AttU-Net, WPN models respectively. The models were trained with all 12 input features.

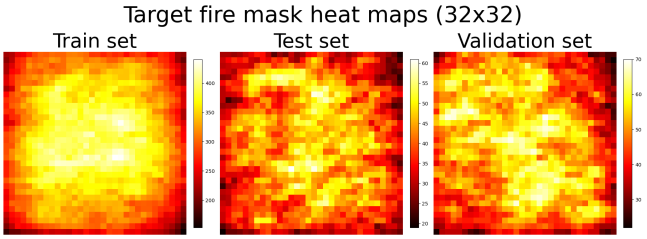

Figure 7: Heat maps showing the distribution of fire in randomly cropped 32x32 target fire masks from the training, testing, and validation sets, respectively.

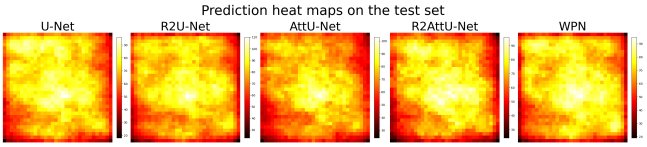

Figure 8: Heat maps showing the distribution of fire predictions for the randomly cropped 32x32 test set from the U-Net, R2U-Net, AttU-Net, R2AttU-Net, and WPN models respectively. The models were trained with all 12 input features.

B RESULTS ON ONLY NEW FIRE PIXELS

In [15], they performed an experiment where the previous fire mask was persisted and used as the prediction. Doing this achieved a relatively high precision of 35.7%, but a low recall of 27.3% (F1 score: 0.3094). This made us curious to see how well the models perform at predicting new fire pixels, i.e. pixels that were not on fire in the previous fire mask. The results on only new fire pixels can be seen in Table 4 and Table 5. The 32x32 model that performed the best on the new fire pixels was the standard U-Net model with 3 features achieving an F1 score of 0.268, and the best 64x64 model was the WPN with 12 features achieving an F1 score of 0.310. One interesting thing to point out is that the Recurrent Residual (R2) U-Net models (R2U-Net and R2AttU-Net) performed much worse on the new fire pixels when only using the previous fire mask as their inputs. This seems to suggest that using at least some of the other modalities/features is important when using R2U-Net based models, and perhaps not as much when using the other models.

Models (32x32)	Modalities	Precision	Recall	F1 Score	Total New Fire Preds.
U-Net	12	21.3%	30.5%	0.251	44847
	7	21.8%	31.9%	0.259	45682
	3	24.4%	29.8%	**0.268**	38236
	Prev. fire mask	22.6%	28.4%	0.252	39248
R2U-Net	12	18.7%	32.1%	0.236	53649
	7	22.7%	29.2%	0.255	40242
	3	20.3%	32.1%	0.249	49329
	Prev. fire mask	27.0%	13.1%	0.177	15199
AttU-Net	12	21.3%	31.3%	0.254	45914
	7	23.1%	30.2%	0.262	40780
	3	25.2%	26.2%	0.257	32483
	Prev. fire mask	22.5%	28.8%	0.252	40077
R2AttU-Net	12	21.2%	30.7%	0.251	45182
	7	22.2%	29.3%	0.253	41176
	3	22.5%	28.7%	0.252	39964
	Prev. fire mask	24.3%	20.2%	0.220	25941
WPN	12	21.9%	31.0%	0.256	44264
	7	20.7%	33.7%	0.257	50707
	3	24.4%	27.1%	0.256	34739
	Prev. fire mask	22.1%	30.7%	0.257	43437

Table 4: Results for the 32x32 models on only new fire pixels (pixels that weren't on fire in the previous fire mask), evaluated using the F1 score

Models (64x64)	Modalities	Precision	Recall	F1 Score	Total New Fire Preds.
U-Net	12	23.9%	31.3%	0.271	58230
	7	27.8%	30.5%	0.291	48962
	3	23.7%	43.8%	0.308	82278
	Prev. fire mask	28.5%	29.6%	0.290	46352
R2U-Net	12	21.2%	43.3%	0.284	91035
	7	21.1%	43.3%	0.284	91206
	3	25.0%	32.4%	0.282	57581
	Prev. fire mask	20.2%	23.4%	0.217	51530
AttU-Net	12	25.8%	37.8%	0.307	65349
	7	25.5%	38.5%	0.307	67455
	3	26.1%	37.9%	0.309	64772
	Prev. fire mask	21.8%	42.5%	0.288	86741
R2AttU-Net	12	21.3%	44.0%	0.287	91862
	7	24.0%	40.2%	0.300	74770
	3	22.6%	39.6%	0.288	78084
	Prev. fire mask	19.8%	18.7%	0.192	42125
WPN	12	27.8%	35.0%	**0.310**	55924
	7	24.4%	37.1%	0.295	67783
	3	25.7%	36.8%	0.302	63799
	Prev. fire mask	29.2%	30.8%	0.300	46963

Table 5: Results for the 64x64 models on only new fire pixels (pixels that weren't on fire in the previous fire mask), evaluated using the F1 score