DISSERTATION

DEEP LEARNING FOR BIOINFORMATICS SEQUENCES: RNA BASECALLING AND PROTEIN INTERACTIONS

Submitted by

Don Neumann

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2024

Doctoral Committee:

Advisor: Asa Ben-Hur

Ross Beveridge
Nathaniel Blanchard
Anireddy Reddy

ABSTRACT

DEEP LEARNING FOR BIOINFORMATICS SEQUENCES: RNA BASECALLING AND PROTEIN INTERACTIONS

In the interdisciplinary field of bioinformatics, sequence data for biological problems comes in many different forms. This ranges from proteins, to RNA, to the ionic current for a strand of nucleotides from an Oxford Nanopore Technologies sequencing device. This data can be used to elucidate the fundamentals of biological processes on many levels, which can help humanity with everything from drug design to curing disease. All of our research focuses on biological problems encoded as sequences.

The main focus of our research involves Oxford Nanopore Technology sequencing devices which are capable of directly sequencing long read RNA strands as is. We first concentrate on improving the basecalling accuracy for RNA, and have published a paper with a novel architecture achieving state-of-the-art performance. The basecalling architecture uses convolutional blocks, each with progressively larger kernel sizes which improves accuracy for the noisy nature of the data.

We then describe ongoing research into the detection of post-transcriptional RNA modifications in nanopore sequencing data. Building on our basecalling research, we are able to discern modifications with read level resolution. Our work will facilitate research into the detection of N6-methyladeosine (m6A) while also furthering progress in the detection of other post-transcriptional modifications.

Finally, we recount our recently accepted paper regarding protein-protein and host-pathogen interaction prediction. We performed experiments demonstrating faulty experimental design for interaction prediction which have plagued the field, giving the faulty impression the problem has been solved. We then provide reasoning and recommendations for future work.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Asa Ben-Hur, for his support and friendship throughout my PhD. I am very grateful for the opportunity to work on problems in biology which is not only impactful, but has also enabled me to learn a wide range of machine learning methods which are applicable to any domain. I would like to thank Dr. Anireddy Reddy for his help within the biological domain and his easy willingness to explain things. I would also like to thank the rest of my committee, Dr. Ross Beveridge and Dr. Nathaniel Blanchard for their support and ideas.

In addition, I would like to thank the NSF funded transcdisciplinary graduate program GAUSSI for their funding and support. The work reported here was partially supported by a National Science Foundation grant (DGE-1450032). Any opinions, findings, conclusions or recommendations expressed are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Finally, I would like to thank my friend Ibrahim Bouzaid for his help and guidance.

DEDICATION

I would like to dedicate this to my late father, Don Neumann Sr.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

As nature stores the blueprints for life in the form of sequences, discerning, utilizing and learning from these sequences has vast implications to understand the fundamentals of biology. Our research begins by addressing the ability to discern these sequences using new technology from Oxford Nanopore Technologies (ONT), which for the first time, allows us to sequence RNA as-is. Prior technology requires laborious chemical pre-processing which involves an intermediary process which alters the RNA, resulting in information loss. This includes the ability to sequence gene transcripts as nature has created them, without the loss of any molecular modifications to the nucleotides.

While this new technology has great potential, it unfortunately suffers from limitations which results in a lower accuracy in sequencing, as compared to prior technology. The lower accuracy is due to multiple reasons which include the similarity of the raw signal between nucleotides, device chemistry limitations [1], and the fact that the electrical signal is measured in picoamps, making it very noisy. Research has shown that these errors, when compounded, results in a high rate of deletions, mismatches, and insertions. Reported accuracy rates are roughly 85-95 percent, whereas the prior technology is at 99 percent.

We address the accuracy issue in our first published paper by improving the state of art accuracy in basecalling RNA utilizing recent machine learning research from the computer vision domain. Our paper is the only published research addressing RNA basecalling, and improves upon the accuracy when compared to ONT's own basecaller. In addition, one published paper stated our research "suggest a promising direction for species-specific basecallers" [1]. The necessary technological background is detailed in Section 2.2, and our research is covered in Chapter 4.

In addition, Oxford Nanopore's new technology is the only device capable of sequencing RNA as-is which allows us to theoretically detect post-transcriptional, or biomolecular modifications, to the RNA nucleotides. While we know that the most prevalent post-transcriptional modification

affects many metabolic processes including gene expression and mRNA decay, its role has yet to be fully determined. This is especially true of the over 170 other post transcriptional modifications which have been discovered, many of which we know nothing about. Thus, the ability to detect post-transcriptional modifications will have a huge impact on the understanding of biology. We address this in Chapter 5 with our research which has the capability to detect and pinpoint the most abundant post-transcriptional modification m6A, while simultaneously basecalling raw ONT device signals. An introduction to post-transcriptional modifications is covered in Section 2.3.

Finally, we conclude with a chapter on experimental design parameters for determining protein-protein interactions from their amino acid sequences. Unfortunately this domain is hampered with problematic ground truth data due to multiple reasons, the first being the inability to experimentally determine negative interactions. In addition, the experiments to determine positive interactions are costly, laborious, and suffer from a high false positive rate. This has led to flawed design parameters for many recently published papers which utilize an erroneous method to generate negative interactions which makes the problem easier. These flawed experimental designs heave led to papers showing high accuracy which gives the impression the protein interaction prediction has made significant progress. Unfortunately, this is not the case which we demonstrate in our research which also proposes design parameters for datasets and experiments. This is detailed in Chapter 6.

1.1 Overview of chapters

The following chapters cover the necessary biological background and related work. Chapter 2 delves into the biological background and relevant technology. Chapter 3 covers the deep learning concepts which have been used in our work.

Continuing, Chapter 4 showcases our first published paper which improved RNA basecalling accuracy for Oxford Nanopore Technologies (ONT) sequencing devices, beating the accuracy of ONT's own basecaller. Chapter 5 presents our current research, which builds on the research in Chapter 4, and can detect and pinpoint m6A which is the most abundant type of post-transcriptional

modification. Chapter 6 then presents published research on proper experimental design parameters for determining whether proteins interact using their amino acid sequences.

Finally, we conclude with Chapter 7 which summarizes our contributions and future work.

Chapter 2

Biological Background and Motivation

The central dogma of biology states that DNA is transcribed to RNA which is then translated into proteins. All three of these fundamental molecular components for life are encoded in sequences. Many of these sequences are incredibly long, such as the human genome, which is comprised of over three billion base pairs with over 20,000 identified genes [2]. There is much to be discovered in how these genes function and interact [3]. All of these molecular sequences have their own defined vocabulary where each item in their respective language represents a different biological molecular structure. Our focus is on RNA and proteins.

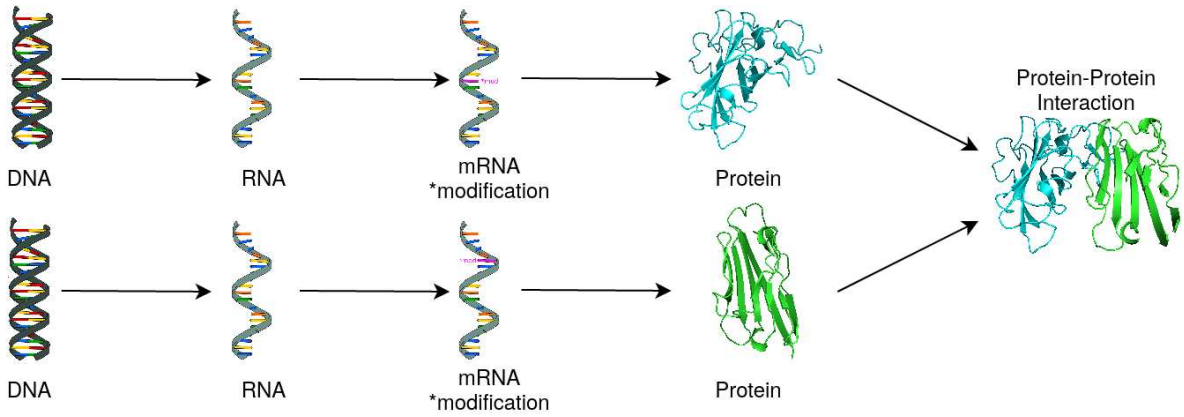


Figure 2.1: DNA is transcribed to RNA, which can be further biochemically modified. RNA is then translated to proteins which may be involved in protein-protein interactions. Parts of figure taken from [4].

The portions of DNA which code for the functions of life are called genes, which through biochemical machinery are transcribed to RNA. We use the word transcribe as the biological process is essentially making a copy with a similar molecular structure to DNA. The RNA is ostensibly used as a coding template for translation into a protein sequence, and the RNA may undergo post-transcriptional modifications such as molecular changes that affect the way RNA functions. As RNA is chemically similar to DNA, the vocabulary is also similar where both are comprised of a 4

letter alphabet, with T for thymine being replace with U for uracil in RNA. We note that the process for determining the sequence of nucleotides in a strand of DNA or RNA is called basecalling.

Post-transcriptional modifications are an important area of study as the epitranscriptome, which covers all of these biomolecular changes to RNA, and its impacts remain largely unknown [5]. There are reportedly over 170 different types of post-transcriptional modifications [6]. In Chapter 5, we will be covering research into the detection of the most abundant modification N6-methyladeosine (m6A) [6], where the adenine nucleotide loses a hydrogen atom and gains a methyl group.

mRNA is then translated by different biochemical machinery into a protein which is a sequence of amino acids. Proteins perform a wide variety of critical jobs within an organism which includes structural, molecular transportation, and regulatory functions [7]. Proteins are comprised of one or more chains of sequences of twenty different amino acids, with their vocabulary shown in table 2.1. The primary structure of a protein is defined by its sequence of amino acids. Ultimately, this sequence determines the proteins three dimensional structure, which determines the proteins function [7].

Table 2.1: Vocabulary for all 20 naturally occurring amino acids.

Amino Acids	Code	Abbreviation	Amino Acids	Code	Abbreviation
Alanine	Ala	A	Leucine	Leu	L
Arginine	Arg	R	Lysine	Lys	K
Asparagine	Asn	N	Methionine	Met	M
Aspartic acid	Asp	D	Phenylalanine	Phe	F
Cysteine	Cys	C	Proline	Pro	P
Glutamine	Gln	Q	Serine	Ser	S
Glutamic acid	Glu	E	Threonine	Thr	T
Glycine	Gly	G	Tryptophan	Trp	W
Histidine	His	H	Tyrosine	Tyr	Y
Isoleucine	Ile	I	Valine	Val	V

2.1 Protein-protein interactions

Proteins rarely act on their own and perform their biological functions by interacting with other proteins through physical contact [8]. As protein-protein interactions (PPI) perform a diverse range of functions within an organism, having a good reference interactome is critical for understanding the fundamental biochemical processes for life. Insight into these interactions can potentially "facilitate therapeutic target identification and novel drug design" [9]. This is especially true with host-pathogen protein interactions, where PPI may be involved in every step of a pathogen's spread of infectious disease [10]. Knowledge about these host-pathogen interactions (HPI) can facilitate inhibition of these interactions, potentially stopping the spread of disease.

While over 200 million protein sequences have been determined and are available in UniProt [11], unfortunately the interaction networks of these proteins are far from complete [12]. This stems from the dynamic and transient nature of many interactions, which make them difficult to detect [8], and the costly and time consuming experimental procedures required to determine whether proteins interact. Due to this incompleteness, there is an entire research area for predicting PPI. In addition, experimental PPI procedures suffer from high false positive rates [13], and lack experimentaly determined negative interactions [14]. These issues are problematic for training prediction models, and appropriate dataset creation requires the generation of negative samples and properly splitting training and test data to prevent information leakage. **We address PPI dataset creation and experimental design parameters to accommodate for these problems in Chapter 6.**

2.2 Oxford Nanopore Technology sequencing

The most prevalently used devices for sequencing RNA are from Illumina which are roughly the size of a desktop [15]. Prior to sequencing, RNA is extracted and chopped into smaller portions which are then reverse transcribed to complementary DNA (cDNA) which is then sequenced. These fragments of cDNA have a maximum read length ranging from 150 to 300. As the RNA is fragmented during sample preparation, these portions of basecalled cDNA are then stitched

together using computational algorithms. This process has two drawbacks. First, we are only sequencing fragments which require sequence assembly. Second, any post-transcriptional molecular modifications to the RNA are lost during the reverse transcribing process.

There are newer sequencing devices, specifically devices from Oxford Nanopore Technologies, which address both these issues [16]. ONT's smallest sequencer is about the size of a stapler, requires minimal sample preparation, and can perform long read sequencing of both DNA and RNA in real time. This means both DNA and RNA can be sequenced as is without requiring any fragmentation or algorithmic stitching. This also opens the door for the direct study of the epitranscriptome, which includes post-transcriptional modifications.

ONT sequencing devices work by pulling a strand of RNA or DNA through a small hole, or pore, in a synthetic protein which sits on a synthetic polymer membrane inside the device. As the strand is pulled through the pore, the ionic current displacement is read which produces a one dimensional time frequency signal which is similar to an audio signal. This signal is so complex it requires machine learning methods, where the most accurate basecaller uses deep learning, to basecall the signal into the nucleotide sequence. This is extremely difficult given that not only is the signal read in picoamps, the surrounding nucleotides affect the signal of the nucleotide in the pore, the translocation speed of the strand through the pore varies, and nature is very noisy. These problems are visible in Figure 2.2 which displays a plot of a small segment of a single basecalled strand of RNA.

This is unfortunate as ONT's technology is very promising for many reasons, including the capability to rapidly sequence the entire genomes of organisms. While there is ongoing research to solve the accuracy problem, both computationally and biologically as ONT has continuously released new pores [17], basecalling accuracy is still only 85-95 percent due to the challenging nature of the data [18].

ONT basecalling is a sequence to sequence based task, and both the first open source [19] and official basecaller from ONT [20] relied on statistical methods, using hidden Markov models. These models required an additional step prior to basecalling for event segmentation, where further

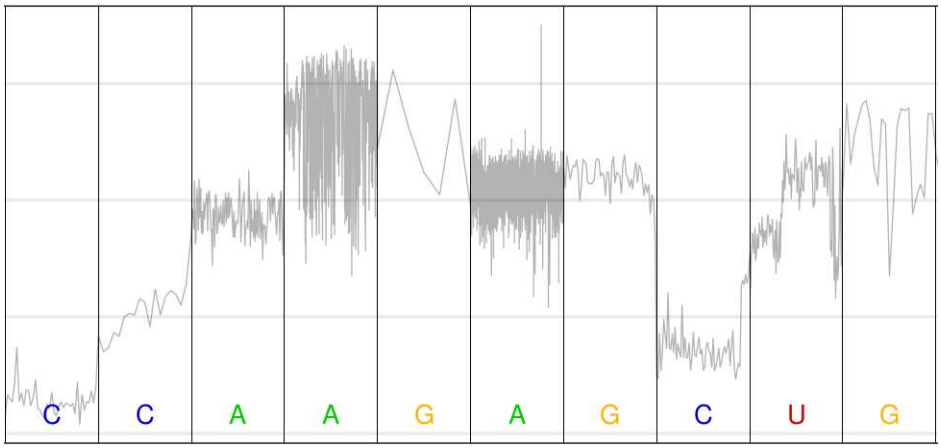


Figure 2.2: Plot of raw ONT signal of a single RNA strand. The variable length and large variance of the signal is visible for each nucleotide, depicting the variable translocation speed and large amount of noise encountered as an RNA strand passes through a pore. Generated with Tombo.

statistical methods are used to detect abrupt changes in the raw signal data. The segmented events are supposed to represent each nucleotide as it passes through a pore. Unfortunately, the accuracy on these early models was still only roughly 70 percent [21]. ONT then shifted to a recurrent neural network with the release of Albacore v2.0.1 [22] which improved the accuracy and eradicated the need for pre-processing with event segmentation. This coincided with the release of the first open source basecaller Chiron [23] which also did not require pre-processing by utilizing both convolutional and recurrent layers. Basecalling has since gone through an evolution of different RNN and convolution based architectures, where even attention [24] has been utilized. Currently, the most accurate basecaller for DNA is ONT's Bonito [25] which is fully convolutional and is still under development. Unfortunately, all of the published open source models have concentrated on basecalling DNA. **We have published the first, and most accurate [1] RNA basecaller, which is fully convolutional, covered in Chapter 4**.

2.3 Post-transcriptional modifications

Detecting post-transcriptional modifications requires laborious experiments such as MeRIP-seq [26] or miCLIP [27] where strands of RNA containing a methylated nucleotide are separated from normal RNA, reverse transcribed to DNA, and sequenced utilizing devices from Illumina.

While miCLIP allows single nucleotide resolution, MeRIP-seq is more widely used due to the simpler nature of the experimental protocol [28]. Unfortunately, MeRIP-seq suffers from a high rate of false positives [29] and does not provide the exact position of the modified base. Instead, the sequences are mapped to a reference genome or transcriptome and the peaks, where the most overlapping occurs, are generally indicative of or near the methylated nucleotide's position. However, this can be problematic where multiple modified nucleotides are close together, resulting in large peaks [30]. The modified "A" nucleotide tends to occur within a particular context, or motif. The context in nucleotide notation is denoted RRACH [31], where R represents A or G, H represents A or C or U, and the center A is methylated. The context is similar across species.

In addition to ongoing work to improve basecalling accuracy, there has been research into detecting post-transcriptional modifications utilizing Oxford Nanopore Technologies devices. Initial research indicated the rate of basecalling errors, at or near modified positions, are higher in a wildtype sample abundant in modifications when compared to a mutant sample, which is deficient in modifications [31]. These errors predominantly occur as mismatches, where the basecalled nucleotide does not match the reference genome, as depicted in Figure 2.3. This research inspired a class of tools, starting with Epinano [31], which utilize these basecalling errors to identify methylated positions. Building on this idea, the Barton Group published a tool called differr [32] which used statistical methods to compare basecalling mismatches between wildtype and mutant samples, producing a list of genomic positions which are likely modified. There have also been other published tools such as Eligos [33] and Drummer [34] which utilize statistical analysis with basecalling errors.

Further research utilizes comparison of the ONT raw signal differential between mutant and wildtype samples. The most notable of these tools is xpore [35] which has shown that the mean ONT signal distribution shifts when comparing mutant and wildtype samples, providing another avenue for modification detection. It is important to note that xpore, along with other signal differential tools, require signal pre-proceessing using a tool like nanopolish [21] or tombo-resquiggle [36]. These tools align basecalled reads to their raw signal using statistical methods.

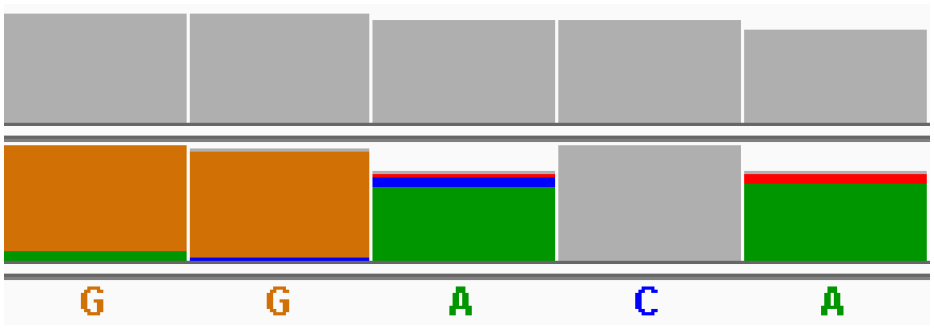


Figure 2.3: Mismatches between mutant sample deficient in m6A (top row) and wildtype sample abundant in m6A (bottom row). Grey represents basecalling errors that have not surpassed the threshold, green represents A, blue represents C, brown represents G, and red represents T. The center A nucleotide is methylated and has a large amount of errors as it is incorrectly basecalled as C or G. The gaps in white spacing is due to deletions. Generated with Integrated Genomics Viewer.

Muliple research tools are available which, while trained on data utilizing sample comparisons, do not require a mutant sample to detect methylated positions. Some examples include MINES [37] and Nanom6A [38]. The most notable, and currently most accurate of these tools [29], is m6anet [39] which uses a neural network based method. **However, none of the currently published tools are capable of direct detection of RNA modifications within a raw ONT signal as they all require some form of pre-processing, or have the capability to simultaneously basecall the reads. This is the focus of our current research and is addressed in Chapter 5.**

Chapter 3

Deep Learning Background

While the field of bioinformatics utilizes many computational and algorithmic approaches, it has long since used machine learning methods to find patterns in biological data. One of the earliest applications of neural networks dates back to 1982, where a perceptron, which is the precursor for neural networks, was used to detect translation initiation sites in mRNA sequences [40]. Since 1982, many new machine learning techniques have been developed which include advances in neural networks. While these advances include new neural network architectures, they also allow us to increase the number of layers which provides the capability of both learning from, and producing, more complex data. It is now common to refer to this field as *deep learning* due to the multiple layers and larger models. Deep learning has produced start-of-the-art solutions for problems from many different domains from image recognition [41] to natural language processing (NLP) [42]. This success has prompted the development of deep learning methods for various bioinformatics problems as there is an overlap in the type of data. This is especially true for NLP, which utilizes sequence data which is also prominent in many areas of biology, and this translation has produced incredible results for problems like protein structure prediction [43].

We will now introduce many of the components which comprise modern sequence-based deep learning architectures. These components cover the way sequences are encoded, and how they are processed with convolution and attention. Figure 3.1 visualizes a generic architecture utilizing all these components and we will be discussing each one in the following sections.

3.1 Encoding of sequences

While sequences can be utilized with many different deep learning methods, we must first address the two different types of sequences and how they are encoded. The first type of sequence is a discrete sequence which has a defined vocabulary. The simplest method to encode a discrete sequence is using a one hot encoding, where each category is represented by a vector of discrete

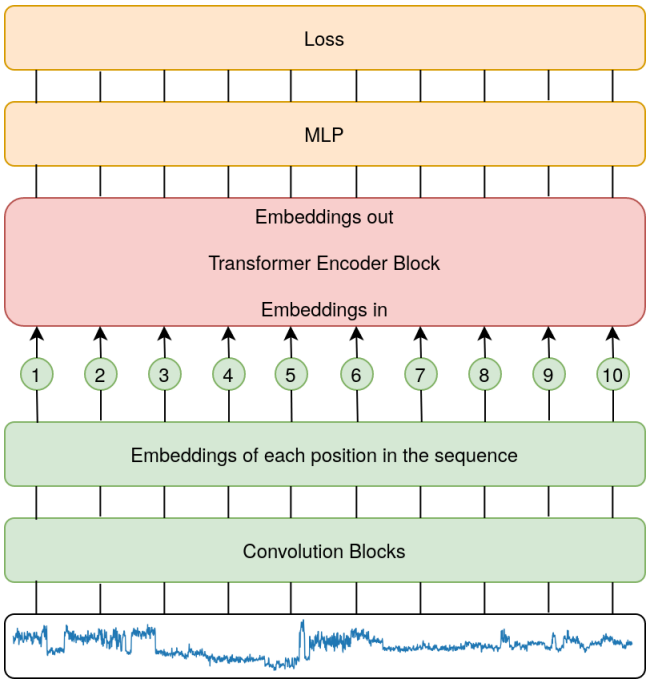


Figure 3.1: Generic sequence based deep learning architecture comprised of convolution and attention. Parts of Figure inspired by [41].

values which is the size of the alphabet. All of the values in each vector are zero except for the defined category which is set to one. Input sequences are then provided to the network as a stack of vectors, where each vector corresponds to the one hot encoding for the defined category in that position of the sequence. The result is similar to a 2 dimensional image, with a matrix the shape of $S \times A$, where S represents the sequence size, and A represents the alphabet size. One hot encodings can be useful with a small alphabet size such as nucleotides, and are used as input to subsequent convolutional layers. However, there are drawbacks to convolutions. First, larger alphabet sizes produce sparse high dimensional vectors which are not computationally efficient. Second, one hot encodings are context independent, and as they are statically defined, they can not learn relationships between sequence elements. These problems can be solved, and one hot encodings can be further improved upon, with embeddings.

Embeddings are typically a low dimensional continuous representation of sparse, high dimensional data. In natural language processing, embeddings can represent and learn semantic similarity between words, where similar words or meanings will have a distance that is closer in the

embedding space [44]. For the amino acids in a protein, which have a vocabulary size of 20, embeddings can capture structural and functional aspects of proteins [45]. As such, embeddings can improve upon one hot encodings as they can also capture systematic relationships within a sequence. In addition, embeddings are often the end result of a neural network and are used for classification, as visualized in Figure 3.1.

The second type of sequence is a one dimensional signal (or time series), where each value in the sequence represents a variable at successive points of time. This is the type of data produced by Oxford Nanopore Technology sequencing devices. While embeddings could theoretically be used with one dimensional signal data, where fixed window segments are fed into a dense layer to produce an embedding, we are unaware of any embedding method used with nanopore sequence data. This is likely due to data's incredibly noisy nature, and as such convolutions have been de-facto standard due to their ability to discern patterns in the noisy signal.

3.2 Convolution

The convolutional neural network operation is inspired by the biological process in the visual cortex [46]. They were initially designed for use in neural networks for visual pattern recognition, where like the visual process, specific patterns will stimulate a response. This is accomplished by performing a dot product between a shared set of weights and the receptive field, or window of the input data. The receptive field is then "slid" across the input data where the same operation is performed against every other position. This turns the input into a feature map which is the output activation's from "sliding" a convolutional filter across the data. This process is visually depicted with a discrete sequence in Figure 3.2 and a time series sequence in Figure 3.3.

Ultimately, convolution allows a neural network to discover and learn patterns in the input data which are translation equivariant, which means the position of the learned pattern can be located anywhere in the input data. While the first layer of convolution will detect simple patterns, like edges in an image, these are combined through locality, where neighboring features are more likely

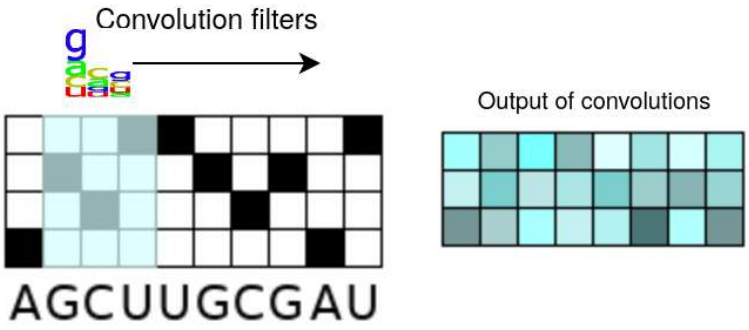


Figure 3.2: Visual depiction of convolution over one a hot encoded RNA sequence. The convolution filters are "slid" across the sequence producing an output for each position.

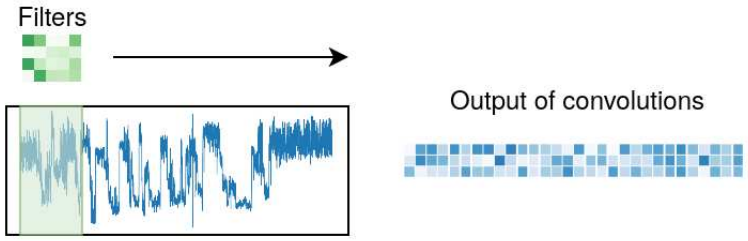


Figure 3.3: Visual depiction of convolution over a one dimensional signal. The convolution filters are "slid" across the sequence producing an output for each position.

to affect each other. Successive convolutions layers will then use this information to discern larger, more complex patterns, as visualized with the stack of convolutional blocks in Figure 3.1.

A convolution is comprised of 3 components: The input data, the kernel (or filter), and the output which is a feature map. Convolutions are applicable in single or multiple dimensions as well as to both signals and discrete data. For convolution over sequence with inputs as one hot encodings which are 2 dimensions, the kernel would be typically be of size $w \times a$, where w is the window size, and a is the alphabet size. In the context of biological sequence data, the one hot encoding would represent the nucleotides in RNA or amino acids in a protein. This kernel is slid along the input data, where at each position, a dot product is computed between the kernel's weights and the input's data in the receptive field. The output is then accumulated into a feature map called a channel. For convolutions over sequence with inputs as one dimensional signal data, the kernel would also be of 1 dimension which would be the window size w. Convolutions typically use

multiple kernels which each produce their own channel, so each kernel corresponds to a different feature detector or learned pattern. Translation equivariance, where the translation of input equals a corresponding translation to the output, is important as each channel is able to discern different patterns in the biological sequence data.

3.3 Attention

The attention mechanism originated with the introduction of transformers in 2017 [47] and is now state-of-the-art for many NLP tasks [48] [42]. This new formulation solved the following problems in recurrent neural networks: parallelization, long range interactions, and is now used in a wide range of different problem domains [45] [41] [49]. While these solutions are a major step, attention mechanisms and transformers require large amounts of data for generalization, as demonstrated in the visual transformer [41]. The default attention mechanism also constructs a quadratic size attention matrix, which requires a large amount of memory that increases with the sequence lengths.

G	A	A	A	G	A	C	A	A	T	G	T	A	A	A	A	C	T
0.0	0.0	0.0	0.0	0.2	0.4	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 3.4: Visual depiction of attention values of modifications identified in an RNA sequence. The middle GAC of the AGACA sequence is highlighted as methylated.

The attention mechanism is applicable to any type of sequence, from the words in a sentence or the amino acids in a protein, to one dimensional signals. It is specifically designed to learn what elements in sequences are related, or what a network should attend to, hence the term *attention*. As we are only interested in the relationship between elements of a single sequence, we will focus on *self-attention*. We can see attention visually depicted in Figure 3.4 where the three nucleotides GAC in the k-mer AGACA are identified with higher attention values, indicating the middle A is modified. This demonstrates the attention mechanism has learned to attend to the relevant portions of the sequence.

Each element in a sequence has its own embedding which can be pre-trained or learnable, and in self-attention we construct matrices used in the attention operation, namely the query Q, key K, and value V matrices from which the attention matrix is constructed:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{3.1}$$

where d_k is the dimensionality of the input embedding space. The product of the query Q matrix and the key K matrix transposed produces a matrix of dot products, which computes the similarity between all combinations of the input embeddings. The softmax function is then applied, producing a probability distribution for each row, where each row represents a single embeddings relation to every other embedding in the sequence. This is then multiplied by the value matrix V which, in effect, recombines the related vectors based on their weighting, into a single vector, which is the end result of the attention operation. This process is visually depicted in Figure 3.5. We note that because of the matrix of dot products, computing attention values has quadratic memory and runtime requirements.

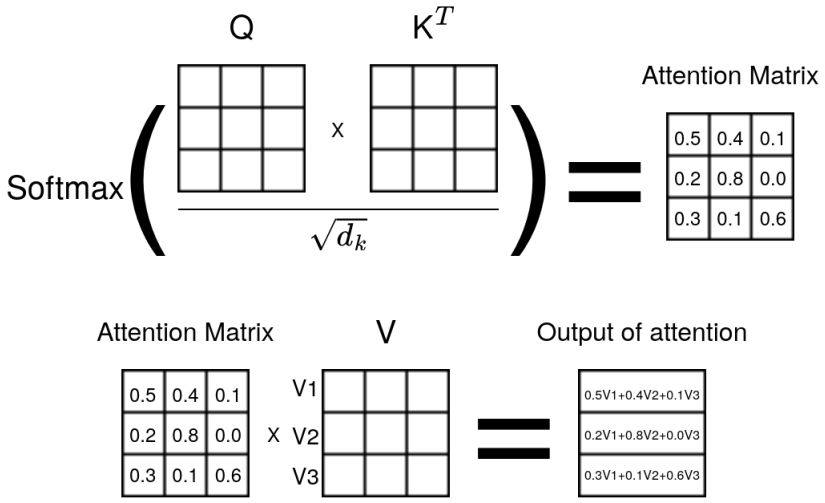


Figure 3.5: Visual depiction of the attention process. The product of the query and key matrix transposed, which is scaled by the square root of the dimensionality of the key matrix, produces an attention matrix. The product of the attention matrix and the values matrix results in a weighted recombination of the values vectors.

An attention mechanism typically uses multiple attention heads, where each head learns to concentrate on different aspects of the information contained in the embeddings. Each attention head i uses a dense layer, denoted W_i^Q, W_i^K, and W_i^V, to project the information from the embedding space into a lower dimensional subspace. After the attention operation, the resulting attention heads are concatenated:

$$\begin{aligned} MultiHeadAttention(Q, K, V) &= Concat(head_1, ..., head_h) \\ \text{where } head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{3.2}$$

The attention operation is typically followed by 2 dense layers (MLP), the first of which expands the size of the concatenated heads, and is followed by an activation function such as GELU [50]. The second dense layer reduces the expansion back to the original embeddings dimensionality and is again followed by a activation function. We can think of this as combining each head's subspace back into one vector space. Residual connections are placed around both the multi-head attention and MLP sublayers. Layer normalization [51] is used for normalization before both the attention and MLP operations. This series of operations is also known as a *transformer encoder block* [47] and is visualized in Figure 3.6. Ultimately, a transformer encoder block produces an embedding, or feature representation, for every element in the sequence which represent the learned information and relations. We note that attention mechanisms are effective for both classification and sequence-to-sequence problems, visualized in our generic sequence architecture shown in Figure 3.1.

As previously stated, the input can be any type of sequence, where in our case, it is the features from prior convolutional layers which have learned the important elements in a one dimensional signal. The attention mechanism can then be trained to discern between the difference in a signal containing a methylated nucleotide, and a non-methylated nucleotide. Even further, we can use multi-task learning to combine both classification and sequence-to-sequence loss functions. This allows us to utilize a pre-pended embedding to classify whether this sequence has a methylated nu-

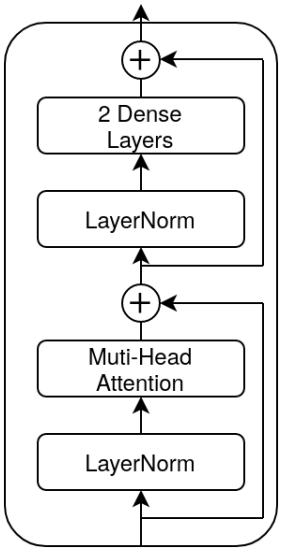


Figure 3.6: Transformer encoder block. A multi-headed attention operation is followed by 2 dense layers to recombine each heads subspace back to a single vector space, where layer normalization is used before both operations which are surrounded by residual connections. Figure inspired by [47].

cleotide which indicates whether we should look at the attention map which allows us to determine the position of said nucleotide. This is detailed in our research in Chapter 4.

Chapter 4

RODAN: a fully convolutional architecture for basecalling nanopore RNA sequencing data

4.1 Introduction

Oxford Nanopore sequencing presents an opportunity for advancements in genomics, transcriptomics, and epitranscriptomics because of its ability to directly sequence a DNA or RNA strand without requiring amplification, producing long reads that can help identify splice isofroms unambiguously, determine poly(A) length, and can potentially capture information on base modifications [52, 53]. Sequencing of both DNA and RNA using this technology occurs by passing nucleotide strands through a synthetic protein pore that straddles a membrane, and recording the resulting current across the membrane. The technology has been developing at a rapid pace since its release in 2014 based on its capabilities for generating long DNA reads and the more recent application to direct RNA sequencing [53]. However, while the technology offers many advantages over other long and short read technologies, it is unfortunately hampered by high error rates [52].

Decoding the current generated by a nucleotide strand as it passes through the pore is a challenging task. This is due to several factors [52]. First, the signal associated with each nucleotide passing through the pore is affected by its surrounding nucleotides (typically, two on each side). Second, the speed of a strand translocating through the pore varies. Therefore, the resulting signal produced by a polymer is a one dimensional sequence of real numbers where each nucleotide is represented by a variable number of sequence values. We will denote this as the *samples per base* associated with a nucleotide. And finally, the electrical signal measured in picoamps, is very noisy. All these factors makes basecalling highly error prone [18, 52].

Improving basecalling accuracy has been the focus of several recent research papers. The earliest approaches were based on recurrent neural networks such as Chiron [23], DeepNano [54]

and Oxford Nanopore Technologies' (ONT) Albacore, which was replaced with Guppy. However, the current trend has been towards fully convolutional architectures such as in ONT's development basecaller Bonito [25] which is based on Nvidia's speech recognition network Quartznet [55]. Further attempts have been made utilizing attention mechanisms as in SACall [24]. While Bonito has improved DNA basecalling accuracy slightly, there is still much room for improvement before ONT's technology can match the accuracy of Illumina sequencing. These improvements are likely to come in both basecalling and the technology itself. Despite all this recent research, most of it focused on DNA data, with little attention to basecalling of RNA data. RNA is sampled by the pore at 70 bases per second (bps), compared to 450 bps for DNA, leading to different signal characteristics, and requiring basecalling methods specifically trained for this data. In this paper we introduce RODAN: **R**NA nan**O**pore **D**ecoding with convolution**A**l **N**etworks, a fully convolutional architecture that achieves state-of-the-art performance on transcriptome data from multiple species including animals and plants. Full implementation is provided on the github repository of this project at https://github.com/biodlab/RODAN.

4.2 Methods

4.2.1 Architecture and training

We propose a fully convolutional basecalling neural network which takes an intuitive approach to decoding the signal generated by ONT direct RNA sequencing. Convolutional networks have emerged as an important technique for working with noisy one dimensional signals [56], and are therefore a good approach for decoding the signal generated by ONT data. A convolutional network scans the input signal with a set of filters or kernels that make up its convolutional layer (see Figure 4.1). Each of these kernels computes a function over a small segment of the signal; the results of the local computation are then fed to the next layer of computation, and can be stacked to create multi-layer "deep" models.

There is much recent research on the design of deep convolutional networks, and the architecture for RODAN is inspired by Google's EfficientNet [57]. EfficientNet improved the state of the

art in image classification while simultaneously reducing the number of model parameters by an order of magnitude. The RODAN architecture is composed of 22 convolutional blocks, contains roughly 10 million parameters, and utilizes a similar convolutional block structure (see Figure 4.1). RODAN gradually incorporates surrounding information for each position in the signal by increasing the kernel size with each successive convolutional block. By increasing the kernel size, we expand the window size to incorporate surrounding signal information which accommodates for the variable samples per base and gathers the necessary information from neighboring nucleotides for accurate decoding. We note that the training and validation set were sampled from data where roughly 83% of all chunks of 4096 values ranged between 20 and 70 samples per base.

4.2.2 Architecture details

The RODAN architecture is composed of 22 convolutional blocks and contains around 10M parameters. The first block is a regular convolution with a kernel size of 3 which acts as a "smoothing" layer to denoise the signal. The smoothing convolution is followed by a squeeze and excitation block. The remaining blocks, as depicted in Figure 4.1, are composed of separable convolutions, where depthwise convolution is followed by a squeeze and excitation block, then a pointwise convolution [58]. All squeeze and excitation blocks forego using a reduction ratio and instead reduce to a fixed size of 32. When the number of channels increases between layers, the convolutional block also includes a pointwise expansion to increase the number of channels before the depthwise convolution. Each convolution operation is followed by a batchnorm and is passed through the Mish activation function [59]. The Mish activation function also replaces ReLU in the squeeze and excitation block. In addition, residual connections have been replaced with ReZero [60] which parameterize's the residual addition.

In our architecture, we increase the number of channels and the kernel sizes used in each layer, up to 768 channels and a kernel size of 100 in the final layer. Its output is then fed to a fully connected layer, followed by a classification layer with a log softmax activation function. The connectionist temporal classification (CTC) loss [61] was used as the objective function for

training the network. We use the Ranger optimizer version 0.1.1 [62], which combines RAdam [63] with lookahead [64], with an initial learning rate of 0.002 and the default weight decay of 0.01. Learning rate decay is utilized at a rate of 0.5 with a scheduler patience of 1 and a threshold of 0.1. Only $1,000$ batches were used for validation. The neural network architecture is detailed in Supplementary Table A.1.

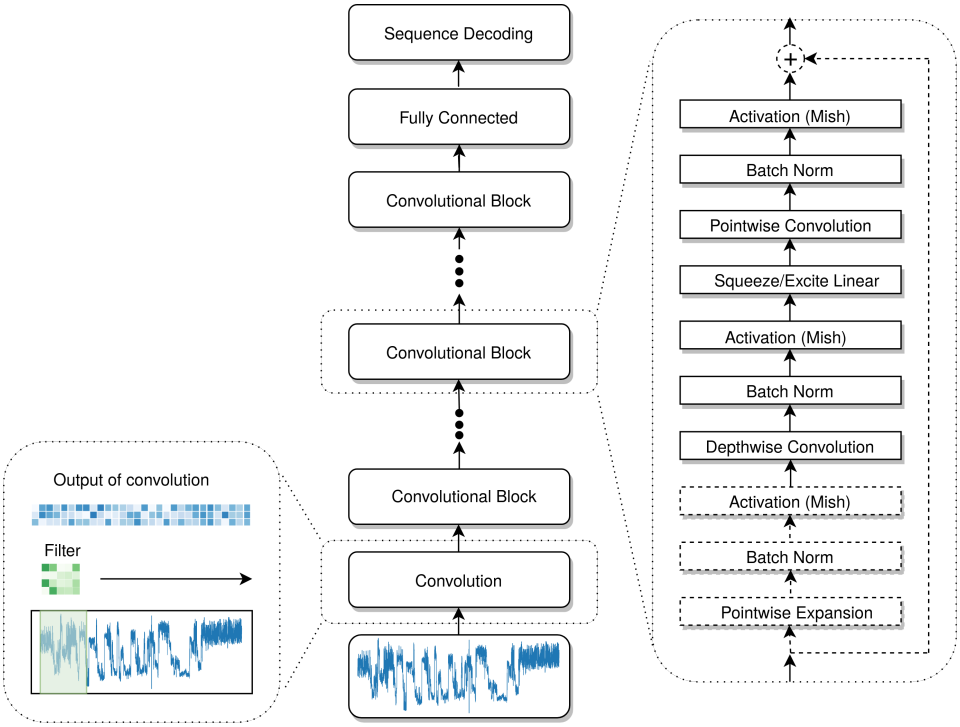


Figure 4.1: The RODAN architecture. The normalized signal is passed through a succession of convolutional blocks which gradually incorporate surrounding information. Each block is composed of several processing steps (convolution, activation, batch normalization etc.), which are standard building blocks in the construction of deep neural networks. The final output is passed through a fully connected layer to produce the decoded sequence of nucleotides.

4.2.3 Model Training

Training of RODAN was performed on an HP Z440 workstation with 6x3.6Ghz dual core processors, 16 GB of RAM, and an Nvidia Titan V GPU with 12 GB of memory. Training was performed using PyTorch version 1.5.1 with the maximum possible batch size of 30 and stopped

after 20 epochs. Label smoothing was also utilized by reweighting the blanks in the CTC sequence with a higher probability of 0.1. The nucleotide vocabulary is then reweighted uniformly at 0.025. Basecalling is performed with a beam search size of 5.

Training of Taiyaki was performed utilizing version 5.0.0 on the same hardware setup. We used the suggested RNA training parameters which are a base layer size of 256, a stride of 10, and number of epochs equal to 10.

4.2.4 Data

Training data. The RNA training data was selected from samples from an in house *Arabidopsis thaliana* wild type which utilized flow cell version R9.4.1, Epinano synthetic constructs (R9.4.1) which contain all possible 5-mers ([31], *Homo Sapiens* (R9.4) from the NA12878 project (BHAM_Run1) [65], *Caenorhabditis elegans* (R9.4) from [66], and *Escherichia coli* (R9.4) from [67].

To generate the Arabidopsis data, total RNA from 17 days-old Arabidopsis thaliana Col-0 seedlings grown on $\frac{1}{2}$ MS at $20°C$ (16/8 hrs light/dark cycle) was isolated using TRIzol reagent and suspended in $160\mu l$ of DEPC-treated water. DNAse treatment was performed by adding $20\mu l$ of 10x DNase buffer and $20\mu l$ RNAse-free DNAseI and incubated for 30 minutes at $37°C$. RNA was then purified using phenol/chloroform. Poly(A)+ mRNA was isolated from about $150\mu g$ of total RNA using the Oligotex Direct mRNA kit (Qiagen). One μg of poly(A)+ RNAs was converted into a library with the Direct RNA Library kit SQK-RNA002 (Oxford Nanopore). The library was sequenced on a SpotON R9.4.1 FLO-MIN106 flowcell, using a GridION x5 sequencer.

All reads were first basecalled with Guppy version 3.4.5 followed by a Tombo version 1.5.1 [68] resquiggle to assess the alignment qualities using the signal matching score and qscore provided by Tombo. The signal matching score (SMS) assesses the quality of the raw current signal against the expected signal, where higher scores indicate lower quality. As Tombo uses a default of 2 for RNA, all reads were filtered with ≤ 2 for the SMS. All reads were filtered with ≥ 11 for qscore, except for the *E. coli* sample which used ≥ 8 for the qscore due to the low quality of the reads. The

remaining reads for each sample were then processed using Oxford Nanopore's research training model Taiyaki according to their instructions [69]. Taiyaki stores the resulting data in an HDF5 file which includes the raw signal data for each read, along with its genomic sequence and alignment positions.

The resulting HDF5 file is comprised of 116,072 reads, 24,370 from Arabidopsis aligned to the Araport11 [70] transcriptome, 29,728 from Epinano synthetic constructs aligned to their released reference [31], 30,048 from *H. Sapiens* (BHAM_Run1) aligned to v33 of the gencode [71] transcriptome, 24,192 from *C. elegans* aligned to the CE11 [72] transcriptome, and 7,734 from *E. coli* aligned to the transcriptome generated from the genome and annotations in the NCBI assembly database [73]. Both the Arabidopsis and *E. Coli* transcriptomes were generated from their respective genomes and gff annotations using the gffread command from cufflinks [74] with the -O option to add non-transcript records.

From this dataset, reads were randomly selected for either training or validation purposes. Each read had a random starting point chosen between 0 and 1024 signal values, and was then segmented into chunks of 4096 values where only chunks with a maximum of 15 samples per base were selected. After a million chunks are selected for training, the remaining reads are then used to select $100,000$ chunks for validation. The raw input signals are normalized by median absolute deviation.

Test data. The test set for measuring the accuracy of our basecaller is comprised of five different samples. These samples originated from studies distinct from those used to generate our training data except for the human data which is taken from a different lab from the Nanopore WGS Consortium's NA12878 project [65]. For each sample, a selection of reads was basecalled with Guppy v4.4.0. Any read which aligned to the mitochondrial genome was discarded. Of the basecalled reads which aligned to each transcriptome, 100,000 were randomly selected for inclusion in the dataset.

The RNA test data is composed from datasets originating from multiple species. Data for *Homo sapiens* (R9.4) was selected from the NA12878 project (BHAM_Run1) [65] and aligned

to v36 of the gencode human transcriptome [71]. *Arabidopsis thaliana* (R9.4) data is the Col-0 wildtype from [32] aligned to the Araport11 [70] transcriptome. *Mus musculus* (R9.4.1) is from [75] and aligned to the vM25 gencode transcriptome. *S. cerevisiae S288C* (R9.4.1) from [33] is aligned to the transcriptome from the NIH genome database [76]. The *Populus trichocarpa* (R9.4.1) from [38] is aligned to the transcriptome generated from the genome and annotations in the NIH assembly database [77]. The Poplar transcriptome, in addition to the Arabidopsis, were generated in the same manner as the transcriptomes for the training data.

4.2.5 Evaluation

Basecallers were evaluated using sequence identity is defined as:

$$accuracy = \frac{M}{M + S + I + D} \tag{4.1}$$

where M is the number of matching bases, S is the number of mismatches, I is the number of insertions, and D is the number of deletions. Two sample t-tests were performed using the scipy stats function ttest_ind comparing the results between RODAN and Guppy, and RODAN and Taiyaki.

4.3 Results and Discussion

We compared RODAN to other available RNA basecallers which include the latest release of ONT's production basecaller Guppy and their research software Taiyaki [78]. Taiyaki was trained with our generated training data. Both Guppy and Taiyaki are based on recurrent neural networks (RNNs). The inherently sequential processing of data with RNNs interferes with modeling long term dependencies and parallelization. Convolutional architectures on the other hand are easily parallelizable.

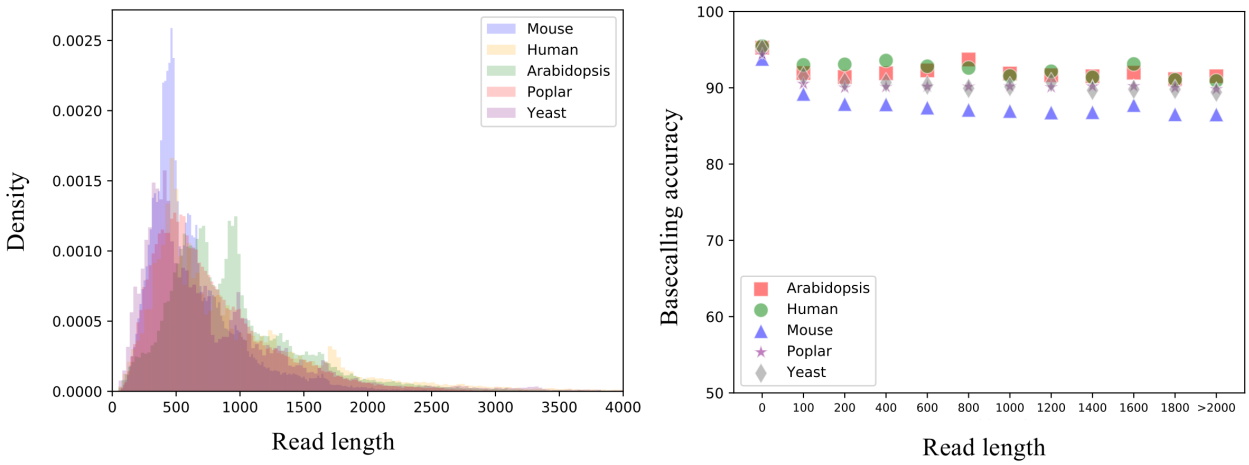
For our evaluation we used the five benchmark datasets described above to assess the accuracy of RODAN across multiple species. Read length distributions across datasets were very similar as seen in Figure 4.2(a). Basecalled reads were aligned with minimap2 2.17-r941 [79] against the

respective transcriptomes [detailed in Methods]. All supplementary alignments were discarded. Basecalling accuracy is shown in Table 4.1, reported as median sequence identity, similarly to other papers reporting basecalling accuracy [18]. We observe that RODAN outperforms Guppy and Taiyaki in all five datasets, with the largest difference in human. The only exception is in yeast, where Guppy was able to match RODAN's performance. All the differences except for RODAN vs Guppy in yeast were highly statistically significant (p-value less than 2.7510^{-157} using a t-test applied as described in Methods). We note that all three basecallers had difficulty with the mouse dataset. This may be the result of not having trained the model on mouse data. To test this hypothesis, we added 24,295 reads of mouse data from [33] to the training set and retrained RODAN with the same configuration. This increased the median accuracy of the mouse from 87.99% to 89.37%. However, it decreased human median accuracy by 1.2% and increased the number of unaligned reads across the remainder of the test data. In poplar, another eukaryote, the model performs well despite not having been trained on data from it. We also report on the total amount of unaligned reads for each basecaller. Taiyaki, which was trained on our generated training dataset, performed slightly better in that regard. We note that the dataset was prepared with Guppy, hence the number of unaligned reads is not applicable.

To obtain more detailed understanding of model performance, we show basecalling accuracy as a function of read length in Figure 4.2(b) and Supplementary Figure A.1. In all datasets we observe a slight decrease in accuracy with read length. We hypothesize that shorter length reads have less of a tendency to form structures that would impede their movement though the pore, leading to more accurate basecalls. In our experiments, Guppy ran around 7x faster than RODAN. This is to be expected since Guppy is optimized production code that is written in C++. Both were run on an HP Z440 workstation with 6x3.6Ghz dual core processors, 16 GB of RAM, and an Nvidia Titan V GPU with 12 GB of memory.

Table 4.1: Basecalling accuracy computed using percent identity and number of unaligned reads across datasets for Guppy 4.4.0, Taiyaki 5.0, and RODAN 1.0. Each dataset contains $100,000$ reads. Only reads alignable by Guppy were used to build each dataset, hence the N/A for unaligned reads. Additional basecalling statistics are provided in Supplementary Table A.2.

Dataset	Basecaller	Median Accuracy	Unaligned
Human [65]	Guppy	90.60	N/A
	Taiyaki	91.16	900
	RODAN	**93.23**	1307
Mouse [75]	Guppy	87.65	N/A
	Taiyaki	86.25	3079
	RODAN	**87.99**	2819
Arabidopsis [32]	Guppy	91.59	N/A
	Taiyaki	91.10	957
	RODAN	**92.89**	1001
Poplar [38]	Guppy	90.16	N/A
	Taiyaki	89.72	1598
	RODAN	**91.11**	1652
Yeast [33]	Guppy	91.35	N/A
	Taiyaki	90.01	2721
	RODAN	**91.41**	3035



(a) (b)

Figure 4.2: Read statistics. For each of the five datasets we show histograms of read length in (a), and basecalling calling accuracy as a function of read length.

4.4 Conclusion

We presented RODAN, an RNA basecaller for nanopore sequence data with state-of-the-art accuracy. Our approach accounts for the varied samples per base and high level of noise inherent to this data with a convolutional architecture that gradually incorporates surrounding information to correctly decode each nucleotide. The software is freely available, and can form the basis for further development. In addition, we have also assembled and released the first comprehensive dataset that can be used to test the accuracy of RNA basecallers in future research [80]. To our chagrin, many published studies do not release raw ONT fast5 data which is crucial to method development and re-analysis of data. We hope this trend improves in the future.

Chapter 5

MOTHRA: Detecting modified nucleotides with nanopore direct RNA sequencing data

5.1 Introduction

While the ONT platform has the capacity to directly sequence RNA, it also potentially provides the ability to detect post-transcriptional modifications to nucleotides. This opens the door to studying the epitransciptome, which is the study of biochemical alterations to RNA, and is largely an unknown [81]. Modification detection is possible by determining the mean signal distribution of a wildtype sample, abundant in modifications, which shifts when compared to a mutant sample which is deficient in modifications, as depicted in Figure 5.2. ONT has published multiple tools with this initial capability for both DNA and RNA which includes Remora [82] which focuses on DNA, and Tombo [68] which performs statistical tests for comparisons to expected current levels as well as sample comparisons.

Prior research indicates basecalling modified nucleotides results in higher amounts of mismatches and deletions, along with lower base quality [31] when aligned to a reference genome after being basecalled with ONT's official basecaller Guppy. Following this, the Barton Group published a tool called differr [32] which utilized statistical tests to compare mismatches between mutant and wildtype samples, helping isolate biochemical modifications. Differ can be utilized in conjunction with previously published research containing published lists of experimentally determined methylated sites to help isolate likely biochemical modifications. Other tools which rely on statistical analysis of basecalling errors include Eligos [33] and Drummer [34].

More recent research relies on statistical or machine learning methods used on segmented events derived from Tombo or nanopolish [21] which determine position and raw signal information. This includes MINES [37] and nanom6A [38] which utilize Tombo events, where xpore [35]

and m6anet [39] use nanopolish. As the signal of each nucleotide is affected by its surrounding nucleotides, the raw signal data is essentially a 5-mer which presents $4^5 = 1024$ possible current levels. First, a raw ONT read is basecalled. Nanopolish then uses this basecalled sequence to align these events based on a model of expected current levels, including the mean and standard deviation of 5-mers released by ONT and available in the nanopolish software [21]. The event (basecalled nucleotide) alignment is performed using a hidden Markov model with the Viterbi algorithm. It is important to note that events can be of different lengths due to the variable translocation speed of strands through the pore, and sometimes strands will be "stuck", further complicating the interpretation of the raw signal.

We are focusing on detecting m6A directly from raw ONT reads without requiring any preprocessing or additional tools. The goal is to provide a tool that when run on experimental data will produce a list of likely modified genomic positions and reads while simultaneously basecalling the reads. This will significantly reduce the time and resources required for post-transcriptional modification detection, while further accelerating research into the biological effects of m6A and facilitating other modification detection.

5.2 Data

Isolating methylated nucleotides is not only difficult, but error prone. The difficulty is due to the multiple ONT device issues previously stated, which also results in the reported signal variance for each k-mer in ONT's signal models being quite large. The variance leads to a large overlap in signal mean distributions between methylated and non-methylated reads. These problems are further compounded by the inadequate basecalling accuracy. In addition, while both Tombo and nanopolish can perform nucleotide event alignments to the raw signals, we have found these alignments are rarely in agreement. Even further, the fact that only a fraction of RNA strands will be methylated [83] makes isolation even more difficult. As such, training a deep learning model to detect methylation in raw reads requires a large amount of experimentally verified data, in conjunction with prior published research with definitive experimental results to reliably isolate positions

in reads which are modified. Given these problems, we have designed a lengthy data preparation process for model training depicted in Figure 5.1.

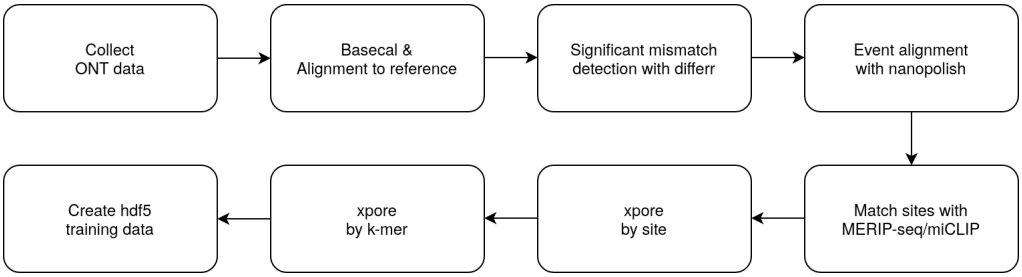


Figure 5.1: Data preparation process starts with collecting the ONT data and ends with generating training data.

We downloaded raw ONT data from prior published research, which was basecalled with Guppy, and aligned with nanopolish against the reference transcriptome. Differr was then run on the data, comparing wildtype and mutant samples, which produces a list of transcriptomic positions with statistically significant mismatch errors. This list of differr sites was then matched against a list of peaks, also obtained from prior research using MeRIP-seq or miCLIP data, where we identified any matching k-mers with an RRACH motif within a window of 5 nucleotide positions surrounding the errors. This is because the mismatch errors are not necessarily on the modified A nucleotide, but can also be in the surrounding nucleotides. The positional information of the matching k-mer was then checked within a window of 10 nucleotide positions against MeRIP-seq or miCLIP peak data. The matching sites were then fed into a modified version of xpore, after aligning the basecalled reads to the reference transcriptome with nanopolish, which tests single sites. While xpore was written to process entire datasets at once, the computation time required to test all sites is excessive, so we created a modification which allows us to test on a per site basis.

As mentioned, xpore operates on nanopolish aligned events by using a multi sample Gaussian mixture model to compare the distributions of means and variances of 5-mers between reads. The comparison requires a mutant sample, deficient in m6A, and a wildtype sample where m6A is

abundant. ONT's previously published 5-mer model of means and variances were used to guide parameter estimations for each model, along with multiple iterations of variational Bayes inference to reach the final conclusion. The mutant and wildtype distributions for each site were then compared with statistical significance testing to produce a list of high confidence methylated sites. A plot of the reads for a single site is presented in Figure 5.2, where we can see the learned Gaussian distribution of both the wildtype and mutant samples.

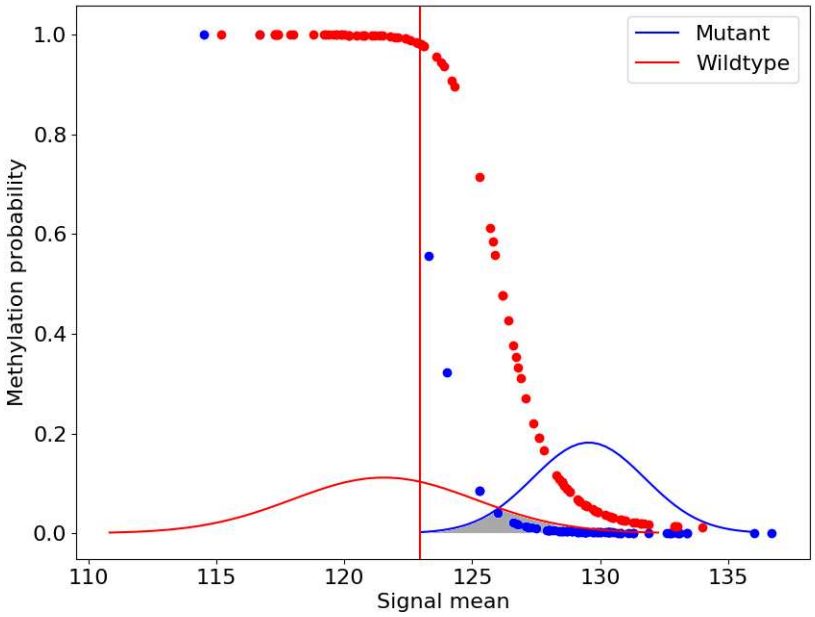


Figure 5.2: Plot of mean signal distributions for methylated transcriptomic position AT1G02150.1:1820 between mutant (blue), deficient in methylation, and wildtype (red), abundant in methylated samples. The shaded area is the overlap between the distributions.

From this list of high confidence methylated sites, we compute the overlap between the distributions and use only sites which share < 0.1 overlap in area under the distributions, with an example distribution overlap shaded grey in Figure 5.3. The reads from these sites are then combined by k-mer, eg AGACT, and run through another modified version of xpore which processes the multiple sites assigned to each k-mer to create a final signal distribution model for each k-mer.

The mean signal distribution of multiple sites are visually depicted for a single k-mer in Figure 5.3.

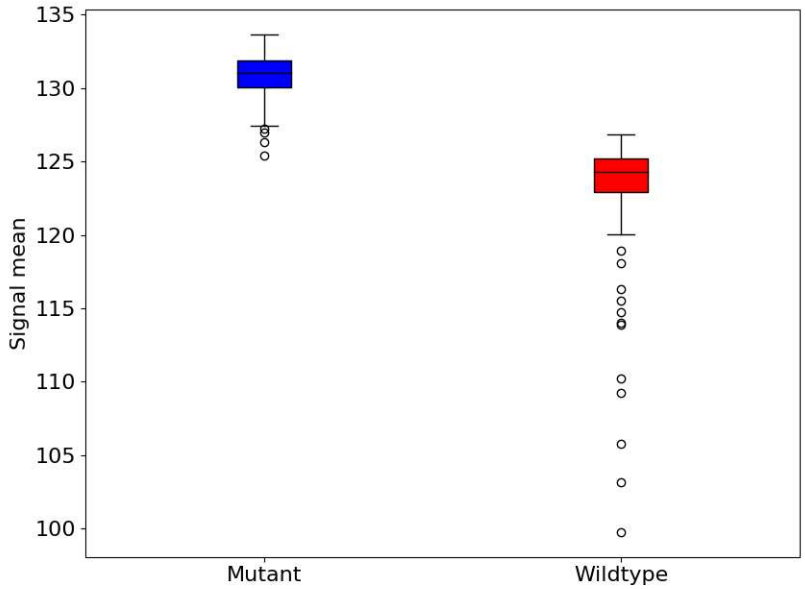


Figure 5.3: Boxplot of signal means across multiple transcriptomic positions for the single k-mer AGACT between mutant (blue), deficient in m6A, and wildtype (red), abundant in m6A.

The final distribution models for each k-mer now has learned parameters consisting of a mean and standard deviation. We then isolate all reads from the mutant sample which are within 3 standard deviations of the unmodified mean, and all reads from the wildtype sample which are < 3 standard deviations from the unmodified mean and have an assigned xpore probability of being modified of ≥ 0.95. The remaining reads were then stored with the nanopolish sequence alignments into the standard hdf5 format used by ONT for data storage.

Using this standard hdf5 format, we can then generate training and validation data. While RODAN uses a chunk size of 4096, we have opted to use a smaller chunk size of 2048 to help isolate methylated positions. The generated training data then consists of chunks of data from methylated and non-methylated site positions, as well as general chunks which are not site related. Chunks with methylated nucleotides are randomly sampled 3 times with different boundaries. Any

chunk which is not within the range of $[20, 70]$ samples per base, as described in the RODAN paper, was discarded. In addition to the nucleotide sequence for each chunk, we assigned a binary classification denoting the chunks methylation status.

5.3 Model

In addition to detecting whether a portion of a read is methylated, we also need to identify a read's nucleotide sequence to discern the modification's genomic position. In our RODAN architecture, we designed a sequence to sequence network which is unsuitable for classification as the final output produces embeddings of the positions within an input sequence. This begs the question, how can we harness the power of RODAN while simultaneously classifying whether a portion of a read is methylated? We could create a summary of the sequence embeddings, similar to image classification models. We opted for the attention mechanism as each of the multiple attention heads can learn to concentrate on different aspects of the sequence information, and the ease with which we can discern the methylated portions of the sequence using the attention maps [84].

Our model is based on a simplified version of the RODAN architecture, where we use multiple RODAN convolutional blocks followed by a standard transformer encoder block comprised of a self-attention layer with 8 heads, as depicted in Figure 5.4. The attention subnetwork then utilizes 2 dense layers which recombines each heads subspace as in a transformer encoder block. In addition to the embeddings from the convolutional network, we prepend a learnable embedding, as in the visual transformer [41], which is separated at the end of the network for classification. The final embeddings are used in for multi-task learning, where the prepended learnable embedding is used in the cross-entropy loss function for the binary classification of methylation, and the remaining embeddings are used the CTC loss function for basecalling the chunks nucleotide sequence.

We pre-train the model without the classification task, where only the CTC function is used for basecalling. The pre-trained model uses Adam for optimization. After pre-training, we add the multi-task classification and use SGD with momentum of 0.9, which is similar to the fine-tuning process for the visual transformer. As of now, we only use AGACA and AGACT k-mer sites for

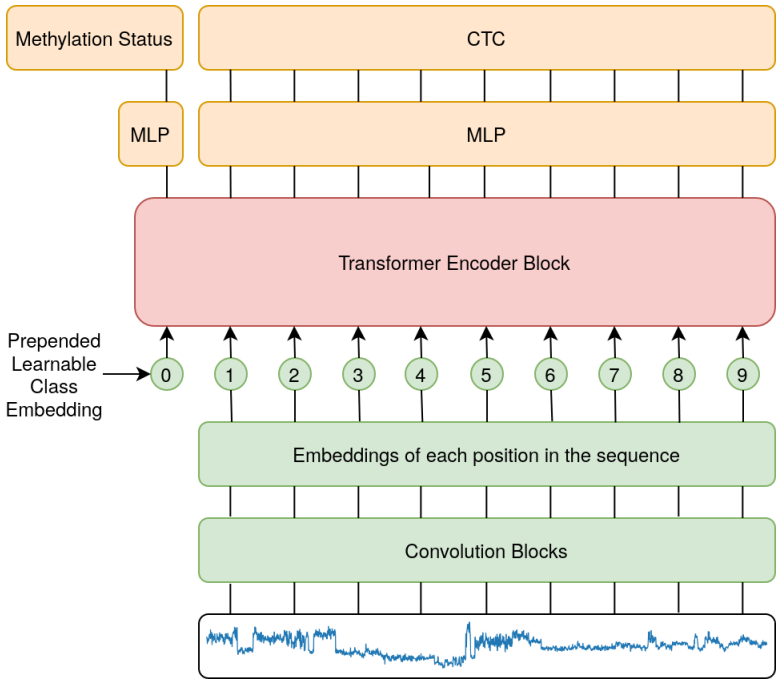


Figure 5.4: Neural network architecture. Raw ONT signal data is processed by multiple RODAN convolutional blocks and the resulting embeddings, along with a pre-pended learnable embedding, are fed through a transformer encoder block. The pre-pended learnable embedding is used for classification, while simultaneously the sequence embeddings are fed through the CTC loss function for basecalling. The numbering refers to the positions of the embeddings in the sequence. Parts of figure inspired by [41].

training with two classes for the cross-entropy loss function which indicate a binary methylated or unmethylated position. We are currently achieving a 96.5% AUPR on the validation data which is comprised of 30,215 unmethylated samples and 1,753 methylated samples. The training and validation data was from Arabidopsis samples published in [32].

G	A	A	A	G	A	C	A	A	T	G	T	A	A	A	A	C	T
0.0	0.0	0.0	0.0	0.2	0.4	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 5.5: Visual depiction of the attention values from the pre-pended embedding in the attention matrix from head 8. The middle GAC of the AGACA sequence which is highlighted is methylated.

We can determine the position of modified nucleotides by analyzing the attention values from our model during inference. As attention head 8 has learned to identify the modified nucleotides, we use the row which corresponds to the pre-pended embedding from the attention matrix of the

appropriate head. This row contains the probability distribution, or attention values, which are depicted in Figure 5.5. We can see the center GAC highlighted in the AGACA k-mer indicating the A nucleotide is methylated. The attention matrix from head 8 is also shown in Figure 5.6 where the highlighted columns also refer to the GAC nucleotides. We used this method to determine the genomic positions of sites in our test data.

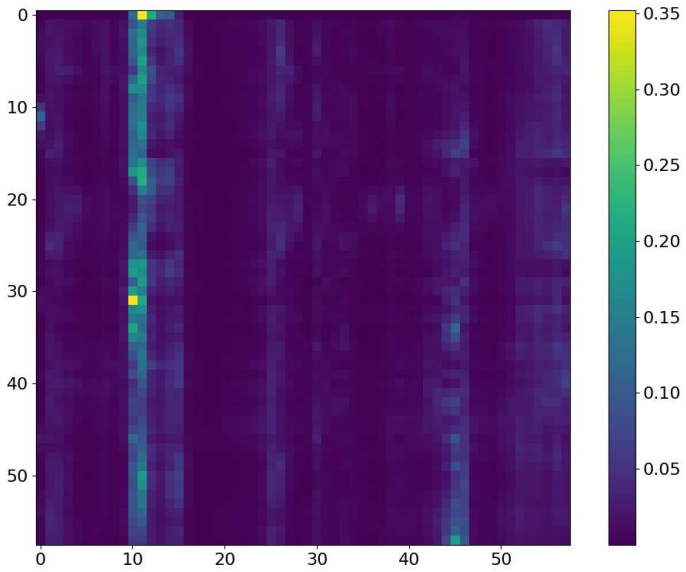


Figure 5.6: Heatmap of the attention matrix from head 8. The high probability columns on the left side corresponds to the GAC in a methylated AGACA k-mer.

The test data for our model was raw nanopore sequencing data from a different Arabidopsis wildtype sample released in [85]. As there is no ground truth available to report an AUPR, we focus on precision which is defined as the methylated sites detected by MOTHRA, found in the combination of three MeRIP-seq datasets published in [86, 87], divided by the total number of methylated sites MOTHRA detected. We plot the precision in Figure 5.7 at increasing prediction thresholds taken from the classifier.

The precision increases from 0.39 to 0.65 with the prediction threshold when limited to AGACA/AGACT sites, as visible from the blue line in our plot. This shows that the more confident MOTHRA's predictions are, the greater our model's precision. We can further improve MOTHRA's precision by only using sites which have more than one (>1) methylated reads. Filtering sites with

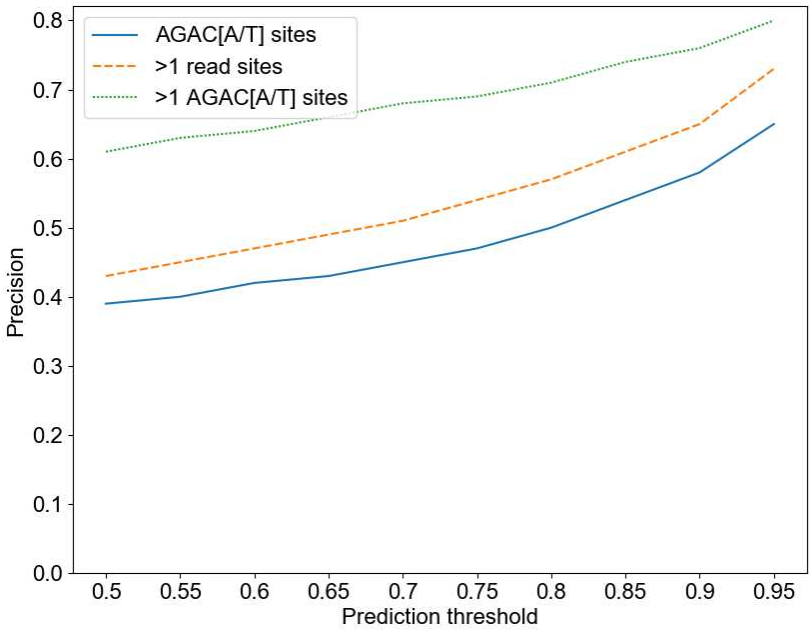


Figure 5.7: Precision of methylated sites, detected by MOTHRA, calculated as a function of the prediction threshold. The blue line plots the precision of all AGACA/AGACT sites, the green is all AGACA/AGACT filtered with >1 methylated reads, and the orange is all sites filtered with >1 methylated reads regardless of k-mer.

>1 reads allows us to remove false positives which are unavoidable due to the noisy nature of the signal which results in a high mean signal variance. When applying this filter to AGACA/AGACT sites, the preicision increases from 0.61 to 0.8 as visible with the green line. We further detail the AGACA/AGACT >1 results with the venn diagram in Figure 5.8 which showcases the level of overlap of MOTHRA and MeRIP-seq sites.

While we explicitly trained our model with only AGACA and AGACT k-mers, there is often other unmethylated and methylated k-mers located close to the intended k-mer's in our training data. As a result, MOTHRA coincidentally learned to identify a limited number of other methylated k-mers as well. When applying the >1 read filter, the precision for all detected sites increases from 0.43 to 0.73 as visible with the orange line. This highlights how MOTHRA was able to distinguish methylated k-mers it was not explicitly trained for. Our results clearly demonstrates our research is successful and needs further work.

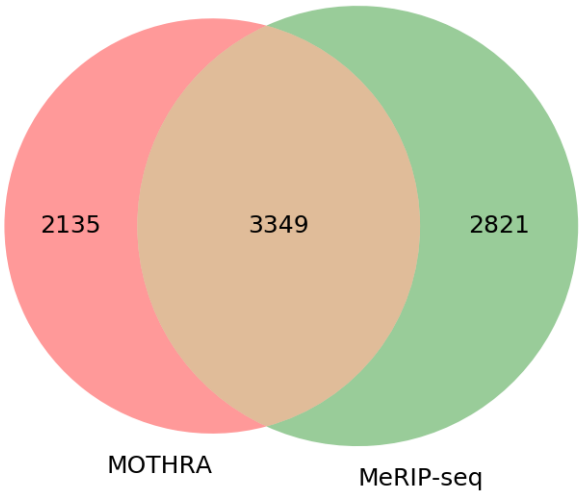


Figure 5.8: Overlap of AGACA/AGACT methylated sites detected by MOTHRA with >1 read which were found in MeRIP-seq data.

5.4 Future work

While we are able to discern the position of modified nucleotides as shown in Figure 5.5, and locate not only AGACA/AGACT but other k-mers coincidentally, there remains a significant amount of time consuming work to do. As previously mentioned, nanopolish and Taiyaki event alignments are rarely in agreement. As such, we plan to re-align all our training data with Taiyaki to achieve the basecalling accuracy reported in RODAN, and then re-run xpore on the new event alignments. This is a lengthy process as the time required for data preparation is exorbitant. We expect this to also provide a more accurate view of methylation events, as discerned by xpore. This will allow us to generate new training examples which are more closely aligned with ONT's own processes.

In addition, we plan to use multi-label classification, where each class represents a different modified k-mer. This will allow us to not only identify the type-of k-mer within a chunk of data, but also detect and pinpoint multiple k-mers simultaneously. The new classification function will use Asymmetric Loss [88] which is the current state-of-the-art for multi-label classification.

Chapter 6

On the choice of negative examples for prediction of host-pathogen protein interactions

6.1 Introduction

Prediction of protein-protein interactions (PPIs), and more recently host-pathogen interactions (HPIs) is a very active area of research in computational biology [89, 90]. Most of the work in this area focuses on prediction of interactions from sequence, especially using deep learning techniques. Some recent publications reported highly accurate prediction results *from sequence alone* that caught our attention [91–93]. As long-time practitioners of machine learning in this area, we approach such results with a healthy dose of skepticism. What could be the cause of such high accuracy? In this paper we focus on one issue related to the choice of negative examples that keeps showing up in various guises.

While databases of PPIs and HPIs are abundant and provide curated information on protein interactions, finding reliable examples of non-interacting proteins is more of a challenge. The Negatome database is one such resource [14]; however, the number of interactions in it is very limited and much smaller than the number of experimentally determined interactions, and does not cover HPIs. In the absence of gold-standard non-interacting proteins, some researchers have chosen to constrain their negative examples in various ways—either by protein localization, justified by the fact that proteins that reside in different cellular compartments are less likely to interact [94] or by constraining the similarity of negative examples to known positive examples [95]. These approaches produce more reliable negative examples than the alternative of choosing random pairs of proteins that are not known to interact, reducing the number of false negatives. However, PPI networks are expected to be very sparse, and therefore the false negative rate for the random pairs method of choosing negative examples is expected to be very small [96]. And as we have discussed

elsewhere [96], the bias introduced by choosing negative examples according to their localization makes the problem easier, inflating prediction performance. Yet another way to introduce a bias on the choice of negative examples is to use proteins with low degrees in the interaction network, since these are less likely to interact with a viral protein of interest [97].

Eid et. al [95] suggested that while PPI networks are indeed sparse, HPI networks are less likely to be so. On the basis of this hypothesis they proposed to choose negative examples by constraining their similarity to positive examples. More specifically, if a host protein is part of the positive set, negative examples of similar host proteins are excluded, since they constitute potential interactions. As we describe below, this is a very effective way of making the prediction problem easier, and indeed provides improved performance. This was demonstrated by Eid et al. and shown here using current deep learning methods. However, this practice is wrong from a machine learning perspective, and we argue that its performance is not expected to hold for real data.

Although some researchers have rightfully shunned the technique of similarity-constrained negative example selection [93, 98], this practice remains present in the field of HPI prediction [91,92,99–103] and also in PPI prediction [104], necessitating this paper to alert researchers to this issue. We have also observed the use of similarity based choice of negative examples in other sequence-based prediction problems such as anti-microbial peptide prediction [105]. We note that the related practice of using cellular compartment to bias the choice of negative examples is also still occasionally being used [106]. The very high accuracy reported in some of the publications cited above may create the wrong impression regarding the accuracy of predicting HPIs from sequence, and it is important that as method developers we be aware of all the potential pitfalls in designing our machine learning experiments.

6.2 Results and Discussion

To demonstrate the effect of using similarity-based sampling on HPI prediction accuracy we implemented the strategy proposed by [95] and created training and test sets characterized by a threshold of the maximum allowed sequence similarity between host proteins that participate in

the training and tests sets (see details in the Methods section). In addition to the original Support Vector Machine (SVM) model of Eid et. al, we applied this strategy to a selection of deep learning models that were developed for PPI and HPI prediction. Model performance was assessed using five fold cross validation for varying sequence similarity thresholds for datasets constructed using two collections of positive examples: the dataset used in Eid et. al, and a larger dataset generated using the latest version of the Host-Pathogen Interaction Database (HPIDB). Results are shown in Figure 6.1. The general trend for all the methods is that performance as measured by the area under the precision recall curve (AUPR) decreases as the similarity threshold increases. For low values of the similarity threshold, i.e. when the distinction between proteins in the training and test sets is extremely well pronounced all the methods achieve close to perfect accuracy, even the simple SVM-based method that uses trimer composition of the two proteins to represent the data. As the similarity threshold increases, the problem becomes more difficult as test set proteins are allowed to become more similar to proteins in the training set. In this regime, the SVM performs at a level that is not much better than a random classifier. The situation is described in Figure 6.2: for a high similarity threshold, the sampling produces what are essentially random pairs that are not known to interact, and the two classes can overlap. As the similarity threshold decreases, the two classes are pushed further apart, making the problem increasingly easy to solve. If this is done just on the training set as in [107], this is appropriate; however, when done on examples on the test set, it makes the test set easy by construction, providing the user with a false sense of success. In-fact, in related work, we have shown that negative examples chosen by constraining sequence similarity does not generalize as well as random pairs for the problem of protein-compound interaction prediction based on an independent test set that uses negative examples chosen as pairs that have low binding affinity [108]. Some authors choose to use similarity-constrained negative examples only in the training set [107]. This way of using similarity-constrained negative examples is not problematic, since there is no information leakage between the training and test sets. However, we suspect that the reduced label noise is not sufficient to compensate for the resulting difference in the distribution of training and test set, and would result in lower prediction accuracy.

It is worth noting that PIPR, which is the most sophisticated deep learning method among those tested is able to maintain a reasonable level of accuracy even for random pairing, and is the most responsive to even low deviations from random sampling. All the other methods required more help in terms of the separation between train and test sets in order to achieve high accuracy.

6.3 Conclusion

In this paper, we discussed pitfalls in the selection of negative examples for host-pathogen and protein-protein interactions. There are other issues that come into play when designing machine learning experiments in this domain. While our focus was on negative example selection, there are multiple issues that are relevant for the choice of positive examples as well: data from experimental methods such as yeast-two-hybrid are known to have a sizable fraction of false positives, and it is common practice to select positive examples by choosing interactions that have been assigned a high confidence score [109]. Another issue is whether to include in the test set interactions for host or pathogen proteins that are present in the training set: if a protein is present in the training set, either as a host or pathogen protein, the classifier is better able to make accurate predictions. So, a naive cross-validation procedure like we have used here provides accuracy estimates that may over-estimate performance if the user is interested in performance over proteins that were unseen by the classifier. This has been discussed by [109, 110] in the context of protein-protein interactions. A common evaluation procedure in HPI prediction is to test the method on novel pathogens for which no data is present in the training set. This captures a likely use case where we wish to obtain potential interactions for an emerging pathogen whose interactions are yet to be studied in the lab. The final issue we would like to mention is class imbalance. Since host-pathogen interaction networks are expected to be sparse, the number of negative examples is expected to be much larger than the number of positive examples, leading to highly imbalanced classification problem. This has impact on the expected classification performance as demonstrated in a recent publication on PPI prediction [13]. Unlike the area under the ROC curve which is invariant to class imbalance, more realistic measures like the area under the precision-recall curve are strongly

affected by class imbalance. In summary, we call upon authors to be aware of these issues and exercise good experiment design that provides valid indication of the method's performance in the real world.

6.4 Methods

6.4.1 Models

The models we selected for our experiments cover a wide variety of sequence based published machine learning methods for HPI prediction from simple methods like the SVM from the original Denovo paper [95] and the single layer convolutional methods DeepViral [98] and DeepTrio [111], to more complex methods like PIPR [9] . In our work we used the original Denovo SVM method [95] as a baseline. The model represents a pair of protein sequences in terms of their k-mer composition vectors normalized to unit vectors and concatenated, to which a Gaussian kernel is applied. Our implementation uses scikit-learn [112] SVM implementation after verifying it produced the same results on their original datasets, and uses 3-mers in a reduced amino-acid alphabet as in the original publication [95].

We also chose a selection of sequence-based deep learning methods of varying complexity. The simplest, DeepViral [98], is a fully convolutional network which uses a single convolutional layer composed of eight different convolutional modules executed in parallel, with convolution applied independently to each protein and concatenated. In our implementation we removed the dropout on the convolutional layer, as we found the model performs much better without it. This is the the sequence-only variant of DeepViral, for a fair comparison with the other methods. Each sequence is one hot encoded and the models were trained for 30 epochs.

PIPR [9] is a more elaborate deep learning architecture for protein-protein interaction prediction comprised of multiple layers of convolution and gated recurrent units. PIPR encodes each amino acid using a vector that combines amino acid composition in a reduced seven dimensional space obtained by clustering amino acids by their properties [113] with a set of features generated using the word2vec skip-gram model which represents the co-occurence of amino acids. The skip-

gram model was trained on $8,000$ sequences from the STRING protein-protein interaction network database [114]. We trained the models for 100 epochs as in the original publication.

We also used DeepTrio [111], a deep learning PPI prediction fully convolutional model which is comprised of 33 convolutional modules executed in parallel on the input sequence. The sequences are one hot encoded and the models were trained for 50 epochs.

All methods used a batch size of 256 with cross entropy loss, and were originally written in Keras and translated to PyTorch [115]. Full implementations are provided on the github repository of this project at https://github.com/biodlab/hpi-neg.

6.4.2 Datasets

In our experiments we used datasets parameterized by the maximum allowed sequence similarity between host proteins in the train and test sets with thresholds ranging from 10% (highly constrained examples, allowing only up to 10% similarity) to 100% (no constraint on similarity between the host proteins in the train and test sets). The original Denovo dataset is comprised of 5,445 human-pathogen interactions, with 445 pathogen proteins and 2,340 human protein derived from VirusMentha [116]. These interactions were used to create 10 different datasets with similarity thresholds between 10% and 100%, where sequence similarity is computed using the Needleman-Wunsch algorithm [117]. For complete details of the algorithm we refer the reader to the original publication [95]. In addition to the original Denovo dataset we created a second much larger dataset (Denovo-HPIDB) based on the latest Host-Pathogen Interaction Database (HPIDB) [118]. HPIDB comprises multiple host and pathogen species, with human being the predominant host. All interactions were restricted to human host only which totaled 50,681 interactions between 9,580 human proteins and 5,930 pathogen proteins.

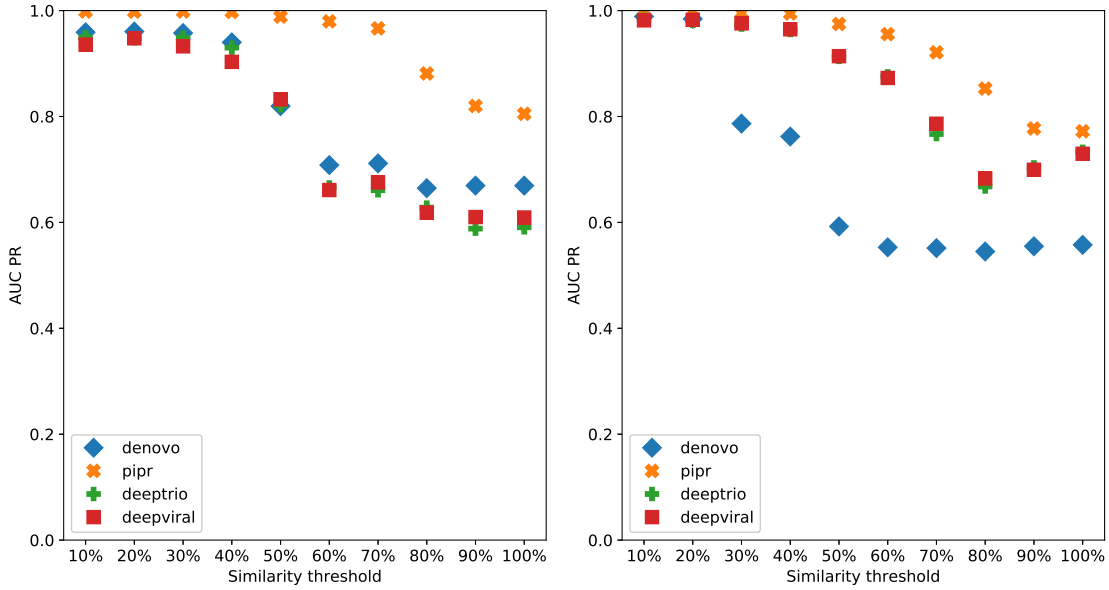


Figure 6.1: Denovo datasets with negative pathogen-host protein pairing by sequence similarity reported as AUCPR for each model, left: originally published Denovo datasets, right: HPIDB based Denovo datasets.

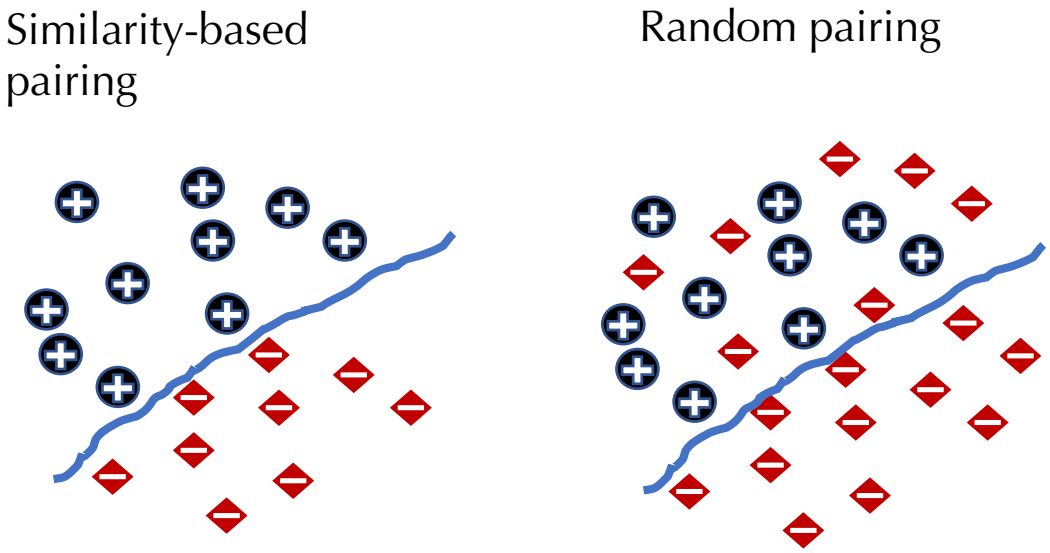


Figure 6.2: The effect of similarity-based selection of negative examples. When using similarity-based selection of negative examples this forces a distinction between positive and negative examples, making the problem much easier to solve.

Chapter 7

Contributions, Conclusions, and Future Work

7.1 Contributions

In this work, we have investigated improving discerning, and learning from biological sequence data. Our contributions are summarized below:

- We achieved state-of-the-art RNA basecalling accuracy utilizing Oxford Nanopore sequencing devices with RODAN, beating ONT's own basecaller. In addition, we released the first publicly available RNA training data for use in future RNA basecalling research. One paper stated RODAN "suggest[s] a promising direction for species-specific basecallers" [1].

- We demonstrate that we can successfully pinpoint the most abundant post-transcriptional modification m6A within raw ONT signal with nucleotide level resolution, while simultaneously basecalling. Our research, which is built upon RODAN, negates the need for complex biological experiments such as MeRIP-seq and miCLIP, as well as the heavy computational requirement for alignment pre-processing for computational methods for detection. We have made the exploration of m6A within the epitranscriptome accessible, and provided a framework for research into detection of other modifications.

- Ongoing research into predicting protein-protein interactions has been hampered with faulty experimental design parameters which include negative sample generation. Our research demonstrates and details these problems, and further provides guidance and design parameters for future work in this field.

7.2 Nanopore basecalling and modification identification

7.2.1 Conclusion

While ONT sequencing devices have the potential to revolutionize the fields of genomics and epitranscriptomics, there are multiple device problems that when compounded, contribute to decreased accuracy. These problems include high signal variance and the short dwell times of some nucleotides as the translocation speed of a strand of RNA varies as it passes through the pore. In addition, signal similarity between nucleotides hinders their differentiation. For instance, the C and U nucleotides are close in signal space and are frequently mistaken for one another [1]. Interestingly, these errors occur in both homopolymers, or regions where the same nucleotide is repeated, and heteropolymers. These errors also affect the detection of methylated nucleotides as we are trying to discern small perturbations in the signal mean which is already highly variable.

Machine learning models can not perfectly accommodate for noisy inconsistent data, and thus will be unable to achieve accuracy comparable to prior technologies such as Illumina. This problem is detailed in a recent ONT RNA sequencing review paper which found the same "systematic error patterns" in RODAN as compared to ONT's own basecaller. They concluded the problems are with ONT's technology and stated the need for "further development of pore chemistry to improve the decoding of raw signal data" [1]. To address these issues, Oxford Nanopore Technology has continued to release new pores [17] as they have improved their devices. Unfortunately, ONT has primarily concentrated on DNA. Ultimately, we may be constrained by the limitations of ONT's current technology, and biologists and researchers will require better device chemistry to improve basecalling accuracy.

7.2.2 Future work

As previously noted, there are over 170 different types of post-transcriptional modifications, most of which we know nothing about. With MOTHRA, we have demonstrated the ability to detect the most abundant type of modification, which opens the door to detect other types of modifications. This allows us entry into an unexplored field, thus providing the potential to revolu-

tionize epitranscriptomics. Instead of requiring laborious biological experiments like MeRIP-seq or miCLIP, or laborious data pre-processing required for current computational methods, we have lowered the barrier to entry with a simple tool which does not require any pre-processing, and will be easily accessible to researchers.

MOTHRA should be extendable to other modifications types, where the main challenge would be preparing the appropriate training data. As any modification should alter the current signal, all we would require is the appropriate mutant and wildtype samples. While any of the aforementioned tools should theoretically be able to discern any modification type, we are unaware if differr has been validated for anything other then m6A. However, similar tools like ELIGOS [33], which also use statistical methods to profile basecalling errors, have been used on 5-methylcytosine (m5C), N1-methyladenine (m1A), and pseudouridine among others. Xpore has been used on m5C [119], and there is a recent pre-print [120] which expands on xpore's method, which reports success in detecting other modification types. We note that many of these and other similar tools, which have attempted to detect other post-transcriptional modifications including m5c, inosine, and pseudouridine [33, 121–123], have publicly released their raw nanopore sequencing data.

7.3 Protein-protein interactions

7.3.1 Conclusion

Contrary to what published research may claim, protein-protein interaction prediction from sequence is far from solved. While one recent paper claims over a 99% AUPR [93] on multiple host-pathogen protein-protein interaction datasets, and another claims over 97% AUPR [104] on a comprehensive dataset comprised of a collection of protein-protein interaction networks from multiple species, their published machine learning models would not work on real world datasets. We demonstrated in our research that these papers employed a faulty dataset generation technique, based on sequence similarity, which made the classification problem much easier.

7.3.2 Future work

We posit that it might be time to give up on solely using sequences as a protein amino acid sequence might not contain enough information for interaction prediction given their complex biochemical structure. Building on this idea, one recent paper used protein structural information from the protein data bank (PDB) and protein language model embeddings with graph attention [124] and achieved a high AUC. This is a promising step. Another recent paper [125] used graph neural networks built on the graph of each protein's molecular structure, which were obtained from DeepMind's AlphaFold [43] structural predictions, with the addition of protein language model embeddings. Their network achieved AUPR's of over 0.95 on two different species datasets. Unfortunately, this paper also generated negative examples based on subcellular localization which we warned about in our paper on dataset generation for PPI.

We can continue to leverage the recent progress on protein structure prediction from AlphaFold. While earlier versions of AlphaFold were generally successful in predicting a single-chain protein structure, they had problems with proteins comprised of multiple chains, or different structures. One paper has successfully modified AlphaFold for PPI prediction and has achieved an AUC of 0.87 for a small dataset of interacting and non-interacting proteins from various species [126]. Given this progress, we will not only be able to employ these newer AlphaFold models in conjunction with PPI datasets to determine whether proteins interact, but also the structure and interface of the resulting interaction.

Bibliography

[1] Wang Liu-Wei, Wiep van der Toorn, Patrick Bohn, Martin Hölzer, Redmond Smyth, and Max von Kleist. Sequencing accuracy and systematic errors of nanopore direct rna sequencing. *bioRxiv*, pages 2023–03, 2023.

[2] National Human Genome Research Institute (NHGRI). Human genome project faq. https://www.genome.gov/human-genome-project/Completion-FAQ, March 2019.

[3] National Academies of Sciences, Engineering, and Medicine and others. *Physics of life*. The National Academies Press, Washington DC, USA, 2022.

[4] File:Difference DNA RNA-DE.svg: Sponk / *translation: Sponk. Comparison of a single-stranded rna and a double-stranded dna with their corresponding nucleobases. https://upload.wikimedia.org/wikipedia/commons/d/db/Difference_DNA_RNA.svg, March 2010. This work is licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/3.0/.

[5] Eva Maria Novoa, Christopher E Mason, and John S Mattick. Charting the unknown epitranscriptome. *Nature Reviews Molecular Cell Biology*, 18(6):339–340, 2017.

[6] David Wiener and Schraga Schwartz. The epitranscriptome beyond m6a. *Nature Reviews Genetics*, 22(2):119–131, 2021.

[7] Jenny Gu and Philip E Bourne. *Structural bioinformatics*, volume 44. John Wiley & Sons, 2009.

[8] Sylwia Struk, Anse Jacobs, Elena Sánchez Martín-Fontecha, Kris Gevaert, Pilar Cubas, and Sofie Goormachtig. Exploring the protein–protein interaction landscape in plants. *Plant, cell & environment*, 42(2):387–409, 2019.

[9] Muhao Chen, Chelsea J-T Ju, Guangyu Zhou, Xuelu Chen, Tianran Zhang, Kai-Wei Chang, Carlo Zaniolo, and Wei Wang. Multifaceted protein–protein interaction prediction based on siamese residual rcnn. *Bioinformatics*, 35(14):i305–i314, 2019.

[10] Daniel P Ryan and Jacqueline M Matthews. Protein–protein interactions in human disease. *Current opinion in structural biology*, 15(4):441–446, 2005.

[11] Uniprot: the universal protein knowledgebase in 2023. *Nucleic Acids Research*, 51(D1):D523–D531, 2023.

[12] Javier De Las Rivas and Celia Fontanillo. Protein–protein interactions essentials: key concepts to building and analyzing interactome networks. *PLoS computational biology*, 6(6):e1000807, 2010.

[13] Brandan Dunham and Madhavi K Ganapathiraju. Benchmark evaluation of protein–protein interaction prediction algorithms. *Molecules*, 27(1):41, 2021.

[14] Philipp Blohm, Goar Frishman, Pawel Smialowski, Florian Goebels, Benedikt Wachinger, Andreas Ruepp, and Dmitrij Frishman. Negatome 2.0: a database of non-interacting proteins derived by literature mining, manual annotation and protein structure analysis. *Nucleic acids research*, 42(D1):D396–D400, 2014.

[15] Rory Stark, Marta Grzelak, and James Hadfield. Rna sequencing: the teenage years. *Nature Reviews Genetics*, 20(11):631–656, 2019.

[16] Daniel Branton, David W Deamer, Andre Marziali, Hagan Bayley, Steven A Benner, Thomas Butler, Massimiliano Di Ventra, Slaven Garaj, Andrew Hibbs, Xiaohua Huang, et al. The potential and challenges of nanopore sequencing. *Nature biotechnology*, 26(10):1146–1153, 2008.

[17] Oxford Nanopore Technologies. Oxford nanopore technology update: Cto clive g brown unveils latest sequencing chemistry with highest performance to date, short fragment mode

and latest methylation performance evaluations. https://nanoporetech.com/about-us/news/oxford-nanopore-technology-update-cto-clive-g-brown-unveils-latest-sequencing, March 2022.

[18] Ryan R Wick, Louise M Judd, and Kathryn E Holt. Performance of neural network basecalling tools for oxford nanopore sequencing. *Genome biology*, 20:1–10, 2019.

[19] Matei David, Lewis Jonathan Dursi, Delia Yao, Paul C Boutros, and Jared T Simpson. Nanocall: an open source basecaller for oxford nanopore sequencing data. *Bioinformatics*, 33(1):49–55, 2017.

[20] Adam Napieralski and Robert Nowak. Basecalling using joint raw and event nanopore data sequence-to-sequence processing. *Sensors*, 22(6):2275, 2022.

[21] Nicholas J Loman, Joshua Quick, and Jared T Simpson. A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nature methods*, 12(8):733–735, 2015.

[22] Oxford Nanopore Technologies. New basecaller now performs 'raw basecalling', for improved sequencing accuracy. https://nanoporetech.com/about-us/news/new-basecaller-now-performs-raw-basecalling-improved-sequencing-accuracy, September 2017.

[23] Haotian Teng, Minh Duc Cao, Michael B Hall, Tania Duarte, Sheng Wang, and Lachlan JM Coin. Chiron: translating nanopore raw signal directly into nucleotide sequence using deep learning. *GigaScience*, 7(5):giy037, 2018.

[24] Neng Huang, Fan Nie, Peng Ni, Feng Luo, and Jianxin Wang. Sacall: a neural network basecaller for oxford nanopore sequencing data based on self-attention mechanism. *IEEE/ACM transactions on computational biology and bioinformatics*, 19(1):614–623, 2020.

[25] Oxford Nanopore Technologies. Nanoporetech/bonito: A pytorch basecaller for oxford nanopore reads. https://github.com/nanoporetech/bonito.

[26] Kate D Meyer, Yogesh Saletore, Paul Zumbo, Olivier Elemento, Christopher E Mason, and Samie R Jaffrey. Comprehensive analysis of mrna methylation reveals enrichment in 3âĂš utrs and near stop codons. *Cell*, 149(7):1635–1646, 2012.

[27] Bastian Linder, Anya V Grozhik, Anthony O Olarerin-George, Cem Meydan, Christopher E Mason, and Samie R Jaffrey. Single-nucleotide-resolution mapping of m6a and m6am throughout the transcriptome. *Nature methods*, 12(8):767–772, 2015.

[28] Alexa BR McIntyre, Nandan S Gokhale, Leandro Cerchietti, Samie R Jaffrey, Stacy M Horner, and Christopher E Mason. Limits in the detection of m6a changes using merip/m6a-seq. *Scientific reports*, 10(1):6590, 2020.

[29] Zhen-Dong Zhong, Ying-Yuan Xie, Hong-Xuan Chen, Ye-Lin Lan, Xue-Hong Liu, Jing-Yun Ji, Fu Wu, Lingmei Jin, Jiekai Chen, Daniel W Mak, et al. Systematic comparison of tools used for m6a mapping from nanopore direct rna sequencing. *Nature Communications*, 14(1):1906, 2023.

[30] Ben R Hawley and Samie R Jaffrey. Transcriptome-wide mapping of m6a and m6am at single-nucleotide resolution using miclip. *Current protocols in molecular biology*, 126(1):e88, 2019.

[31] Huanle Liu, Oguzhan Begik, and Eva Maria Novoa. Epinano: detection of m 6 a rna modifications using oxford nanopore direct rna sequencing. *RNA Modifications: Methods and Protocols*, pages 31–52, 2021.

[32] Matthew T Parker, Katarzyna Knop, Anna V Sherwood, Nicholas J Schurch, Katarzyna Mackinnon, Peter D Gould, Anthony JW Hall, Geoffrey J Barton, and Gordon G Simpson. Nanopore direct rna sequencing maps the complexity of arabidopsis mrna processing and m6a modification. *Elife*, 9:e49658, 2020.

[33] Piroon Jenjaroenpun, Thidathip Wongsurawat, Taylor D Wadley, Trudy M Wassenaar, Jun Liu, Qing Dai, Visanu Wanchai, Nisreen S Akel, Azemat Jamshidi-Parsian, Aime T Franco,

et al. Decoding the epitranscriptional landscape from native rna sequences. *Nucleic acids research*, 49(2):e7–e7, 2021.

[34] Alexander M Price, Katharina E Hayer, Alexa BR McIntyre, Nandan S Gokhale, Jonathan S Abebe, Ashley N Della Fera, Christopher E Mason, Stacy M Horner, Angus C Wilson, Daniel P Depledge, et al. Direct rna sequencing reveals m6a modifications on adenovirus rna are necessary for efficient splicing. *Nature communications*, 11(1):6016, 2020.

[35] Ploy N Pratanwanich, Fei Yao, Ying Chen, Casslynn WQ Koh, Yuk Kei Wan, Christopher Hendra, Polly Poon, Yeek Teck Goh, Phoebe ML Yap, Jing Yuan Chooi, et al. Identification of differential rna modifications from nanopore direct rna sequencing with xpore. *Nature biotechnology*, 39(11):1394–1402, 2021.

[36] Marcus Stoiber, Joshua Quick, Rob Egan, Ji Eun Lee, Susan Celniker, Robert K Neely, Nicholas Loman, Len A Pennacchio, and James Brown. De novo identification of dna modifications enabled by genome-guided nanopore signal processing. *BioRxiv*, page 094672, 2016.

[37] Daniel A Lorenz, Shashank Sathe, Jaclyn M Einstein, and Gene W Yeo. Direct rna sequencing enables m6a detection in endogenous transcript isoforms at base-specific resolution. *Rna*, 26(1):19–28, 2020.

[38] Yubang Gao, Xuqing Liu, Bizhi Wu, Huihui Wang, Feihu Xi, Markus V Kohnen, Anireddy SN Reddy, and Lianfeng Gu. Quantitative profiling of n 6-methyladenosine at single-base resolution in stem-differentiating xylem of populus trichocarpa using nanopore direct rna sequencing. *Genome biology*, 22:1–17, 2021.

[39] Christopher Hendra, Ploy N Pratanwanich, Yuk Kei Wan, WS Sho Goh, Alexandre Thiery, and Jonathan Göke. Detection of m6a from direct rna sequencing using a multiple instance learning framework. *Nature Methods*, 19(12):1590–1598, 2022.

[40] Gary D Stormo, Thomas D Schneider, Larry Gold, and Andrzej Ehrenfeucht. Use of the âĂŸperceptronâĂŹalgorithm to distinguish translational initiation sites in e. coli. *Nucleic acids research*, 10(9):2997–3011, 1982.

[41] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[42] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[43] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[44] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[45] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing, 2021.

[46] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[48] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[49] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[50] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[51] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[52] Shanika L Amarasinghe, Shian Su, Xueyi Dong, Luke Zappia, Matthew E Ritchie, and Quentin Gouil. Opportunities and challenges in long-read sequencing data analysis. *Genome Biology*, 21(1):1–16, 2020.

[53] Daniel R Garalde, Elizabeth A Snell, Daniel Jachimowicz, Botond Sipos, Joseph H Lloyd, Mark Bruce, Nadia Pantic, Tigist Admassu, Phillip James, Anthony Warland, Michael Jordan, Jonah Ciccone, Sabrina Serra, Jemma Keenan, Samuel Martin, Luke E McNeill, Jayne Wallace, Lakmal Jayasinghe, Chris Wright, Javier Blasco, Stephen Young, Denise Brocklebank, Sissel Juul, James Clarke, Heron Andrew J, and Daniel J Turner. Highly parallel direct RNA sequencing on an array of nanopores. *Nature Methods*, 15(3):201, 2018.

[54] Vladimír Boža, Broňa Brejová, and Tomáš Vinař. Deepnano: deep recurrent neural networks for base calling in minion nanopore reads. *PloS one*, 12(6):e0178751, 2017.

[55] Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. Quartznet: Deep automatic speech

recognition with 1d time-channel separable convolutions. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6124–6128. IEEE, 2020.

[56] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[57] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[58] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[59] Diganta Misra. Mish: A self regularized non-monotonic activation function. *arXiv preprint arXiv:1908.08681*, 2019.

[60] Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, pages 1352–1361. PMLR, 2021.

[61] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[62] Ranger optimizer. http://github.com/mpariente/Ranger-Deep-Learning-Optimizer. Accessed 21 February 2021.

[63] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

[64] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019.

[65] Rachael E Workman, Alison D Tang, Paul S Tang, Miten Jain, John R Tyson, Roham Razaghi, Philip C Zuzarte, Timothy Gilpatrick, Alexander Payne, Joshua Quick, et al. Nanopore native RNA sequencing of a human poly (A) transcriptome. *Nature Methods*, 16(12):1297–1305, 2019.

[66] Nathan P Roach, Norah Sadowski, Amelia F Alessi, Winston Timp, James Taylor, and John K Kim. The full-length transcriptome of *C. elegans* using direct RNA sequencing. *Genome Research*, 30(2):299–312, 2020.

[67] Felix Grünberger, Robert Knüppel, Michael Jüttner, Martin Fenk, Andreas Borst, Robert Reichelt, Winfried Hausner, Jörg Soppa, Sébastien Ferreira-Cerca, and Dina Grohmann. Exploring prokaryotic transcription, operon structures, rRNA maturation and modifications using nanopore-based native RNA sequencing. *bioRxiv*, pages 2019–12, 2020.

[68] Oxford Nanopore Technoogies. Nanoporetech/tombo: Tombo is a suite of tools primarily for the identification of modified nucleotides from raw nanopore sequencing data. nanoporetech. http://github.com/nanoporetech/tombo. Accessed 21 February 2021.

[69] Oxford Nanopore Technologies. Taiyaki walk-through. http://github.com/nanoporetech/taiyaki/blob/master/docs/walkthrough.rst. Accessed 21 February 2021.

[70] Chia-Yi Cheng, Vivek Krishnakumar, Agnes P Chan, Françoise Thibaud-Nissen, Seth Schobel, and Christopher D Town. Araport11: a complete reannotation of the arabidopsis thaliana reference genome. *The Plant Journal*, 89(4):789–804, 2017.

[71] Adam Frankish, Mark Diekhans, Anne-Maud Ferreira, Rory Johnson, Irwin Jungreis, Jane Loveland, Jonathan M Mudge, Cristina Sisu, James Wright, Joel Armstrong, et al. Gencode reference annotation for the human and mouse genomes. *Nucleic acids research*, 47(D1):D766–D773, 2019.

[72] Nathan P Roach, Norah Sadowski, Amelia F Alessi, Winston Timp, James Taylor, and John K Kim. The full-length transcriptome of c. elegans using direct rna sequencing. *Genome Research*, 30(2):299–312, 2020.

[73] Asm584v2 - genome - assembly - ncbi. https://www.ncbi.nlm.nih.gov/assembly/GCF_000005845.2. Accessed 21 February 2021.

[74] Cole Trapnell, Brian A Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J van Baren, Steven L Salzberg, Barbara J Wold, and Lior Pachter. Transcript assembly and abundance estimation from rna-seq reveals thousands of new transcripts and switching among isoforms. *Nature biotechnology*, 28(5):511, 2010.

[75] Aleksandra Bilska, Monika Kusio-Kobiałka, Paweł S Krawczyk, Olga Gewartowska, Bartosz Tarkowski, Kamil Kobyłecki, Jakub Gruchota, Ewa Borsuk, Andrzej Dziembowski, and Seweryn Mroczek. B cell humoral response and differentiation is regulated by the non-canonical poly (a) polymerase tent5c. *bioRxiv*, page 686683, 2019.

[76] Saccharomyces cerevisiae s288c (id 15) - genome - ncbi. https://www.ncbi.nlm.nih.gov/genome/15?genome_assembly_id=22535. Accessed 21 February 2021.

[77] Pop_tri_v3 - genome - assembly - ncbi. https://www.ncbi.nlm.nih.gov/assembly/GCF_000002775.4/. Accessed 21 February 2021.

[78] Oxford Nanopore Technologies. Taiyaki research software. http://github.com/nanoporetech/taiyaki. Accessed 21 February 2021.

[79] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[80] Don Neumann, Anireddy S.N. Reddy, and Asa Ben-Hur. Oxford nanopore rna test dataset for rodan. https://doi.org/10.5281/zenodo.4557004. Accessed 1 April 2021.

[81] Michaela Frye, Bryan T Harada, Mikaela Behm, and Chuan He. Rna modifications modulate gene expression during development. *Science*, 361(6409):1346–1349, 2018.

[82] Oxford Nanopore Technologies. Nanoporetech/remora: Methylation/modified base calling separated from basecalling. https://github.com/nanoporetech/remora/.

[83] Miguel Angel Garcia-Campos, Sarit Edelheit, Ursula Toth, Modi Safra, Ran Shachar, Sergey Viukov, Roni Winkler, Ronit Nir, Lior Lasman, Alexander Brandis, et al. Deciphering the âĂIJm6a codeâĂİ via antibody-independent quantitative profiling. *Cell*, 178(3):731–747, 2019.

[84] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

[85] Hang Qin, Liang Ou, Jian Gao, Longxian Chen, Jia-Wei Wang, Pei Hao, and Xuan Li. Dena: training an authentic neural network model using nanopore sequencing data of arabidopsis transcripts for detection and quantification of n6-methyladenosine on rna. *Genome Biology*, 23(1):1–23, 2022.

[86] Lisha Shen, Zhe Liang, Xiaofeng Gu, Ying Chen, Zhi Wei Norman Teo, Xingliang Hou, Weiling Maggie Cai, Peter C Dedon, Lu Liu, and Hao Yu. N6-methyladenosine rna modification regulates shoot stem cell fate in arabidopsis. *Developmental cell*, 38(2):186–200, 2016.

[87] Stephen J Anderson, Marianne C Kramer, Sager J Gosai, Xiang Yu, Lee E Vandivier, Andrew DL Nelson, Zachary D Anderson, Mark A Beilstein, Rupert G Fray, Eric Lyons, et al. N6-methyladenosine inhibits local ribonucleolytic cleavage to stabilize mrnas in arabidopsis. *Cell reports*, 25(5):1146–1157, 2018.

[88] Tal Ridnik, Emanuel Ben-Baruch, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric loss for multi-label classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 82–91, 2021.

[89] Xiaotian Hu, Cong Feng, Tianyi Ling, and Ming Chen. Deep learning frameworks for protein-protein interaction prediction. *Computational and Structural Biotechnology Journal*, 2022.

[90] Xianyi Lian, Xiaodi Yang, Shiping Yang, and Ziding Zhang. Current status and future perspectives of computational studies on human–virus protein–protein interactions. *Briefings in Bioinformatics*, 22(5):bbab029, 2021.

[91] Sho Tsukiyama, Md Mehedi Hasan, Satoshi Fujii, and Hiroyuki Kurata. LSTM-PHV: prediction of human-virus protein–protein interactions by LSTM with word2vec. *Briefings in bioinformatics*, 22(6):bbab228, 2021.

[92] Muhammad Nabeel Asim, Muhammad Ali Ibrahim, Muhammad Imran Malik, Andreas Dengel, and Sheraz Ahmed. LGCA-VHPPI: a local-global residue context aware viral-host protein-protein interaction predictor. *Plos one*, 17(7):e0270275, 2022.

[93] Sumit Madan, Victoria Demina, Marcus Stapf, Oliver Ernst, and Holger Froehlich. Accurate prediction of virus-host protein-protein interactions via a siamese neural network using deep protein sequence embeddings. *bioRxiv*, 2022.

[94] Shawn Martin, Diana Roe, and Jean-Loup Faulon. Predicting protein–protein interactions using signature products. *Bioinformatics*, 21(2):218–226, 2005.

[95] Fatma-Elzahraa Eid, Mahmoud ElHefnawi, and Lenwood S Heath. DeNovo: virus-host sequence-based protein–protein interaction prediction. *Bioinformatics*, 32(8):1144–1150, 2016.

[96] Asa Ben-Hur and William Stafford Noble. Choosing negative examples for the prediction of protein-protein interactions. *BMC bioinformatics*, 7(1):1–6, 2006.

[97] Lopamudra Dey, Sanjay Chakraborty, and Anirban Mukhopadhyay. Machine learning techniques for sequence-based prediction of viral–host interactions between SARS-CoV-2 and human proteins. *Biomedical journal*, 43(5):438–450, 2020.

[98] Wang Liu-Wei, Şenay Kafkas, Jun Chen, Nicholas J Dimonaco, Jesper Tegnér, and Robert Hoehndorf. Deepviral: prediction of novel virus–host interactions from protein sequences and infectious disease phenotypes. *Bioinformatics*, 37(17):2722–2729, 2021.

[99] Abdul Hannan Basit, Wajid Arshad Abbasi, Amina Asif, Sadaf Gull, and Fayyaz Ul Amir Afsar Minhas. Training host-pathogen protein–protein interaction predictors. *Journal of bioinformatics and computational biology*, 16(04):1850014, 2018.

[100] Xiang Zhou, Byungkyu Park, Daesik Choi, and Kyungsook Han. A generalized approach to predicting protein-protein interactions between virus and host. *BMC genomics*, 19(6):69–77, 2018.

[101] Xiaodi Yang, Shiping Yang, Qinmengge Li, Stefan Wuchty, and Ziding Zhang. Prediction of human-virus protein-protein interactions through a sequence embedding-based machine learning method. *Computational and structural biotechnology journal*, 18:153–161, 2020.

[102] Xiaodi Yang, Shiping Yang, Xianyi Lian, Stefan Wuchty, and Ziding Zhang. Transfer learning via multi-scale convolutional neural layers for human–virus protein–protein interaction prediction. *Bioinformatics*, 37(24):4771–4778, 2021.

[103] João Luiz de Lemos Padilha Pitta, Crhisllane Rafaele dos Santos Vasconcelos, Gabriel da Luz Wallau, Túlio de Lima Campos, and Antonio Mauro Rezende. In silico predictions of protein interactions between zika virus and human host. *PeerJ*, 9:e11770, 2021.

[104] Wenqi Chen, Shuang Wang, Tao Song, Xue Li, Peifu Han, and Changnan Gao. Dcse: Double-channel-siamese-ensemble model for protein protein interaction prediction. *BMC genomics*, 23(1):1–14, 2022.

[105] Daniel Veltri, Uday Kamath, and Amarda Shehu. Deep learning improves antimicrobial peptide recognition. *Bioinformatics*, 34(16):2740–2747, 2018.

[106] Tanlin Sun, Bo Zhou, Luhua Lai, and Jianfeng Pei. Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC bioinformatics*, 18(1):1–8, 2017.

[107] Jack Lanchantin, Tom Weingarten, Arshdeep Sekhon, Clint Miller, and Yanjun Qi. Transfer learning for predicting virus-host protein interactions for novel virus sequences. In *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 1–10, 2021.

[108] Adiba Yaseen, Imran Amin, Naeem Akhter, Asa Ben-Hur, and Fayyaz Minhas. Insights into performance evaluation of compound-protein interaction prediction methods. *Bioinformatics*, 38(Supplement 2):ii75–ii81, 09 2022.

[109] Tobias Hamp and Burkhard Rost. Evolutionary profiles improve protein–protein interaction prediction from sequence. *Bioinformatics*, 31(12):1945–1950, 2015.

[110] Yungki Park and Edward M Marcotte. Flaws in evaluation schemes for pair-input computational predictions. *Nature methods*, 9(12):1134–1136, 2012.

[111] Xiaotian Hu, Cong Feng, Yincong Zhou, Andrew Harrison, and Ming Chen. Deeptrio: a ternary prediction system for protein–protein interaction using mask multiple parallel convolutional neural networks. *Bioinformatics*, 38(3):694–702, 2022.

[112] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[113] Juwen Shen, Jian Zhang, Xiaomin Luo, Weiliang Zhu, Kunqian Yu, Kaixian Chen, Yixue Li, and Hualiang Jiang. Predicting protein–protein interactions based only on sequences information. *Proceedings of the National Academy of Sciences*, 104(11):4337–4341, 2007.

[114] Damian Szklarczyk, John H Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos, Nadezhda T Doncheva, Alexander Roth, Peer Bork, et al. The string database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic acids research*, page gkw937, 2016.

[115] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[116] Alberto Calderone, Luana Licata, and Gianni Cesareni. Virusmentha: a new resource for virus-host protein interactions. *Nucleic acids research*, 43(D1):D588–D592, 2015.

[117] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[118] Mais G Ammari, Cathy R Gresham, Fiona M McCarthy, and Bindu Nanduri. Hpidb 2.0: a curated database for host–pathogen interactions. *Database*, 2016, 2016.

[119] P Acera Mateos, AJ Sethi, M Guarnacci, A Ravindran, A Srivastava, J Xu, K Woodward, W Hamilton, J Gao, LM Starrs, et al. Identification of m6a and m5c rna modifications at single-molecule resolution from nanopore sequencing. *bioRxiv (Mar. 2022)*, 14, 2022.

[120] Jannes Spangenberg, Christian Höner Zu Siederdissen, Milena Žarković, Sandra Triebel, Ruben Rose, Christina Martínez Christophersen, Lea Paltzow, Mohsen M Hegab, Anna Wansorra, Akash Srivastava, et al. Magnipore: Prediction of differential single nucleotide changes in the oxford nanopore technologies sequencing signal of sars-cov-2 samples. *bioRxiv*, 2023.

[121] Adrien Leger, Paulo P Amaral, Luca Pandolfini, Charlotte Capitanchik, Federica Capraro, Valentina Miano, Valentina Migliori, Patrick Toolan-Kerr, Theodora Sideri, Anton J Enright,

et al. Rna modifications detection by comparative nanopore direct rna sequencing. *Nature communications*, 12(1):7198, 2021.

[122] Doaa Hassan, Daniel Acevedo, Swapna Vidhur Daulatabad, Quoseena Mir, and Sarath Chandra Janga. Penguin: a tool for predicting pseudouridine sites in direct rna nanopore sequencing data. *Methods*, 203:478–487, 2022.

[123] Sepideh Tavakoli, Mohammad Nabizadeh, Amr Makhamreh, Howard Gamper, Caroline A McCormick, Neda K Rezapour, Ya-Ming Hou, Meni Wanunu, and Sara H Rouhanifard. Semi-quantitative detection of pseudouridine modifications and type i/ii hypermodifications in human mrnas using direct long-read sequencing. *Nature Communications*, 14(1):334, 2023.

[124] Bosheng Song, Xiaoyan Luo, Xiaoli Luo, Yuansheng Liu, Zhangming Niu, and Xiangxiang Zeng. Learning spatial structures of proteins improves protein–protein interaction prediction. *Briefings in bioinformatics*, 23(2):bbab558, 2022.

[125] Kanchan Jha, Sriparna Saha, and Hiteshi Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1):8360, 2022.

[126] Patrick Bryant, Gabriele Pozzati, and Arne Elofsson. Improved prediction of protein-protein interactions using alphafold2. *Nature communications*, 13(1):1265, 2022.

Appendix A

Supplementary Material

A.1 Alignment generation

minimap2 was run with options `-secondary=no -ax map-ont -cs`. The CS tag was used to determine the mismatches, insertions, and deletions to calculate the final accuracies. The accuracy.py script is available in the project's github repository.

A.2 Details of the RODAN architecture

Table A.1: The RODAN network architecture. Kernel denotes the convolution kernel size and stride denotes the kernel step which defaults to 1 unless noted. #Channels denotes the number of kernels utilized. The first block is a normal convolution followed by batchnorm, activation, and a squeeze and excitation layer.

Block	Operator	Kernel / Stride	#Channels
1	Convolution, SQEX	3	256
2	ConvBlock	10	256
3	ConvBlock	10 / 10	256
4	ConvBlock	10	320
5	ConvBlock	15	384
6	ConvBlock	20	448
7	ConvBlock	25	512
8	ConvBlock	30	512
9	ConvBlock	35	512
10	ConvBlock	40	512
11	ConvBlock	45	512
12	ConvBlock	50	512
13	ConvBlock	55	768
14	ConvBlock	60	768
15	ConvBlock	65	768
16	ConvBlock	70	768
17	ConvBlock	75	768
18	ConvBlock	80	768
19	ConvBlock	85	768
20	ConvBlock	90	768
21	ConvBlock	95	768
22	ConvBlock	100	768

A.3 Detailed basecalling accuracy

Table A.2: Detailed basecalling accuracy across datasets for Guppy 4.4.0, Taiyaki 5.0, and RODAN 1.0. Only reads alignable by Guppy were used to build each dataset, hence the N/A for unaligned reads. RODAN (nobeam) refers to a beam search of 1 which is equivalent to greedy decoding. Mismatch, deletion and insertion percentages were computed with respect to the total length of the aligned portions of all reads.

Dataset	Basecaller	Median	Avg.	Unaligned reads	Mis%	Del%	Ins%
Arabidopsis	Guppy	91.59	90.72	N/A	2.32	4.77	2.18
	Taiyaki	91.10	90.37	957	2.28	5.34	2.01
	RODAN	**92.89**	**92.24**	1001	1.90	3.75	2.12
	RODAN (nobeam)	92.51	91.99	1055	1.79	4.76	1.46
Mouse	Guppy	87.65	87.17	N/A	3.88	6.29	2.66
	Taiyaki	86.25	85.97	3079	4.24	7.33	2.46
	RODAN	**87.99**	**87.60**	2819	3.78	6.43	2.19
	RODAN (nobeam)	87.54	87.17	3291	3.61	7.47	1.76
Human	Guppy	90.60	89.87	N/A	2.56	5.35	2.22
	Taiyaki	91.16	90.61	900	2.19	5.27	1.94
	RODAN	**93.23**	**92.62**	1307	1.73	3.52	2.13
	RODAN (nobeam)	92.92	92.45	1086	1.62	4.59	1.34
Yeast	Guppy	91.35	**90.51**	N/A	2.78	4.20	2.51
	Taiyaki	90.01	89.29	2721	3.21	4.71	2.79
	RODAN	**91.41**	90.46	3035	2.91	4.13	2.51
	RODAN (nobeam)	91.11	90.22	3182	2.77	5.21	1.80
Poplar	Guppy	90.16	89.26	N/A	2.95	4.92	2.87
	Taiyaki	89.72	88.90	1598	3.01	5.15	2.94
	RODAN	**91.11**	**90.13**	1652	2.77	4.26	2.84
	RODAN (nobeam)	90.75	89.81	1813	2.64	5.50	2.05

A.4 Accuracy by read length

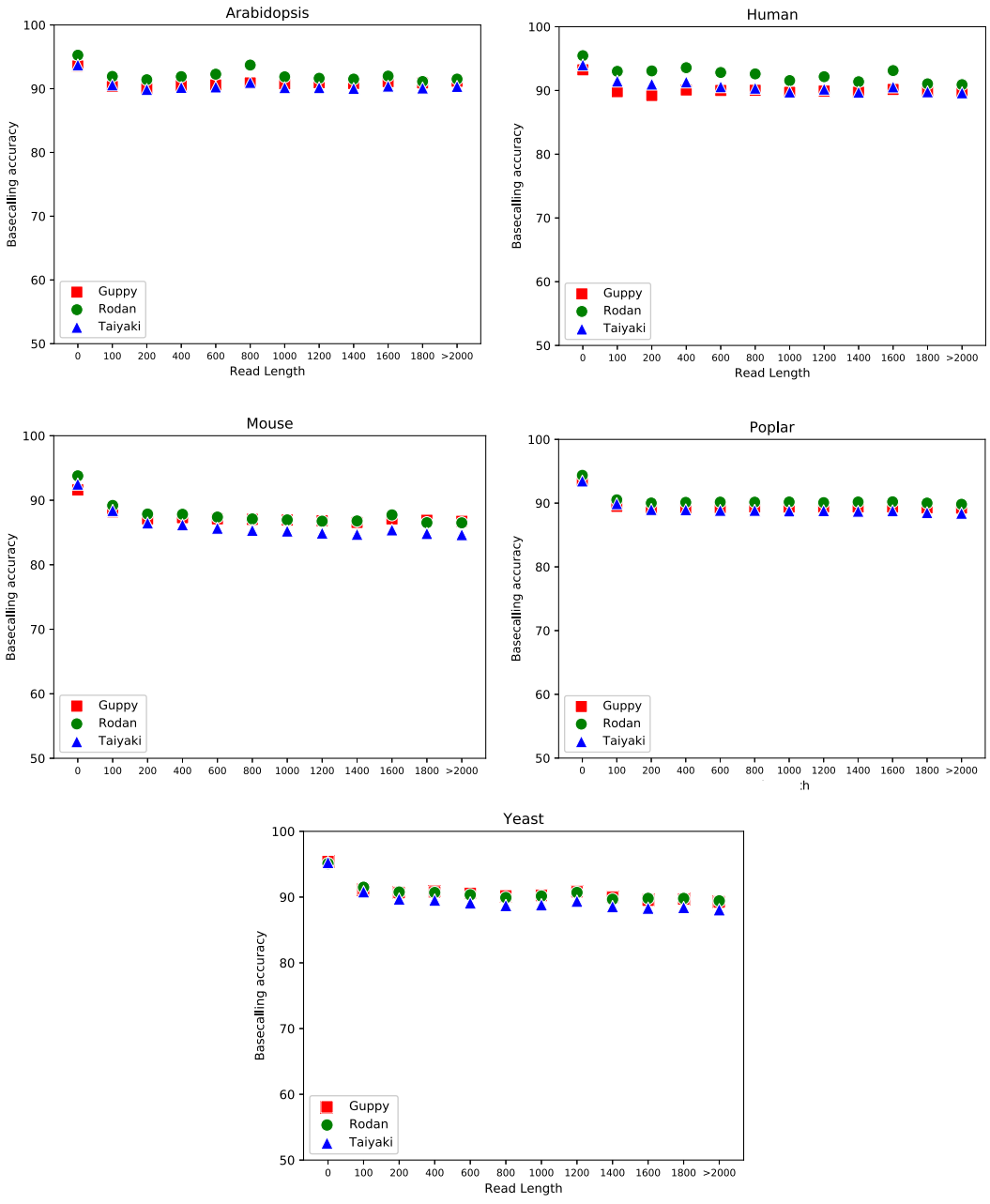


Figure A.1: Accuracy of RODAN, Guppy and Taiyaki as a function of read length across datasets.