FACULTY OF BIOSCIENCE ENGINEERING

DETECTION OF M6A MODIFICATIONS IN NATIVE RNA USING OXFORD NANOPORE TECHNOLOGY

word count: 17.260

Maxime Van Haeverbeke
Student ID: 01405800

Supervisor(s): Prof. dr. Willem Waegeman , dr. ir. Gerben Menschaert
Tutor: Ir. Jim Clauwaert

A dissertation submitted to Ghent University in partial fulfilment of the requirements for the degree of master in Bioscience Engineering.
Academic year: 2018 - 2019

GHENT UNIVERSITY

Ghent, June 7, 2019

The promotors,	The author,
Prof. dr. Willem Waegeman	Maxime Van Haeverbeke
Dr. ir. Gerben Menschaert	

PREFACE

When I chose the subject of this thesis over a year ago, I hoped to learn a great deal about some interesting and trending topics in data science and bioinformatics. I am proud to say that this has indeed been the case. The things I learned and the results of my work are two aspects of what made this thesis an enjoyable experience. A third aspect is the people I had the chance to work with.

I'd like to start by expressing an enormous amount of gratitude to my supervisor and guide through the jungle of deep learning: Ir. Jim Clauwaert. I am very grateful for the many hours you spent filling gaps in my knowledge, helping me overcome the encountered obstacles and critically reviewing the text.

I would also like to thank my promotors, Dr. ir. Gerben Menschaert and Prof. dr. Willem Waegeman, for all their valuable advice and insights, their feedback and, perhaps most importantly, for making this thesis a possibility.

My parents, who have supported me throughout my life, have my deepest gratitude.

Last but not least, I want to thank my twin sister Aurélie, my brother Steven and my friends for all the good times that alternated with the hard work.

CONTENTS

Preface i

Contents v

English abstract vii

Nederlands abstract ix

1 Introduction and outline 1

1.1 Introduction . . . 1

1.2 Thesis outline . . . 1

2 RNA Epigenetics 3

2.1 Introduction . . . 3

2.2 N6-methyladenosine . . . 4

2.3 State-of-the-art m6A detection . . . 5

2.4 The MethylTranscriptome Database . . . 7

3 Bioinformatics and sequencing 9

3.1 Introduction . . . 9

3.2 Sequencing technologies . . . 9

3.2.1 Sanger sequencing . . . 9

3.2.2 Next generation sequencing . . . 10

3.2.3 Illumina . . . 10

3.2.4 Third generation sequencing . . . 10

3.3 Basecalling . . . 13

3.4 Sequence alignment and quality control . . . 14

4 Machine Learning 15

4.1 Introduction . . . 15

4.2 Basics . . . 15

4.2.1 Types of machine learning 15
4.2.2 Parameter optimisation 17
4.2.3 Model capacity and regularisation 18
4.2.4 Model performance evaluation 19
4.3 Deep learning 22
4.3.1 Motivation, advantages and drawbacks 22
4.3.2 Neural network model components 23
4.3.3 Neural network architectures 28

5 M6A detection using basecaller output 31
5.1 Introduction 31
5.2 Nanopore data processing 32
5.3 Data generation 32
5.4 Model architecture 36
5.5 Discussion and model assessment 37

6 Future perspectives and conclusions 43
6.1 Future perspectives 43
6.1.1 Improvement and expansion of the basecaller output model 43
6.1.2 Development of a m6A-augmented basecaller 44
6.2 Conclusions 46

Bibliography 47

Appendix A Direct RNA sequencing runs and quality control 55

Appendix B Distributions of selected data after filtering 57
B.1 Positive set 57
B.2 Negative set 58

Appendix C Manual evaluation of data generation 59

Appendix D bio-informatics commands 61
D.1 Albacore basecalling 61
D.2 Trimmomatic 61
D.3 FastQC 61
D.4 Minimap2 61
D.5 Samtools 61

Appendix E Annotating the raw Nanopore signal **63**

Appendix F Supplementary figures **65**

ENGLISH ABSTRACT

N6-Methyladenosine (m6A) is a frequently occurring post transcriptional mRNA modification that has recently gained much interest as more and more research indicates that it plays a role in the regulation of various biological processes. Its current detection methods are biased, have low throughput and/or do not have a single-nucleotide resolution. Furthermore, they typically require large sequencing machines and labour intensive experiments.

In this work, Oxford Nanopore direct RNA sequencing data is integrated with a database of known m6A sites for the development of a neural network model that is capable of detecting m6A sites in the native mRNA of a sample after direct RNA sequencing.

Keywords: RNA modifications, epitranscriptomics, N6-Methyladenomsine, Oxford Nanopore Technologies, basecalling, deep learning.

NEDERLANDS ABSTRACT

N6-Methyladenosine (m6A) is een vaak voorkomende post transcriptionele modificatie van mRNA dat recent in de belangstelling is gekomen vanwege het toenemend onderzoek dat erop wijst dat het een rol speelt in de regulatie van allerlei biologische processen. De huidige detectiemethoden voor m6A zijn onderhevig aan *bias*, hebben een beperkte schaal en/of een beperkte resolutie. Bovendien vereisen ze typisch grote sequeneringsmachines en arbeidsintensieve experimenten.

In dit werk wordt data afkomstig van *Oxford Nanopore direct RNA sequencing* geintegreerd met een database van gekende m6A sites voor de ontwikkeling van een neuraal netwerk model dat ertoe in staat is m6A sites in ongewijzigd mRNA van een staal te detecteren.

Trefwoorden: RNA modificaties, epitranscriptomics, N6-Methyladenomsine, Oxford Nanopore Technologies, basecalling , deep learning.

LIST OF ABBREVIATIONS

Adam Adaptive momentum.

AI Artificial Intelligence.

ANN Artificial Neural Network.

AUC Area Under the Curve.

BAM Binary Alignment Map.

ChIP Chromatin immunoprecipitation.

CIGAR Concise Idiosyncratic Gapped Alignment Report.

CNN Convolutional Neural Network.

DNA Deoxyribonucleic acid.

FTO Fat mass and Obesity-associated protein.

GTF Gene Transfer Format.

HDF5 Hierarchical Data Format 5.

IGV Integrative Genomics Viewer.

m6A N6-methyladenosine.

MetDB MethylTranscriptome Database.

MLP Multi-Layer Perceptron.

MSE Mean Squared Error.

NGS Next Generation Sequencing.

ONT Oxford Nanopore Technologies.

PCR Polymerase Chain Reaction.

PR Precision Recall.

ReLU Rectified Linear Unit.

RNA Ribonucleic acid.

ROC Receiver Operating Characteristic.

RT Reverse transcriptase.

SAM Sequence Alignment Map.

SGD Stochastic Gradient Descent.

SMRT Single Molecule Realtime sequencing.

WTAP Wilm’s tumor associated protein.

YTHDF YTH Domain Family.

CHAPTER 1

INTRODUCTION AND OUTLINE

1.1 Introduction

A proper regulation of the expression of genes is of paramount importance for the avoidance of many illnesses, including developmental diseases and cancer. Numerous coordinated mechanisms in cells ensure the correct orchestration of gene expression, often targeting the life cycle of RNA molecules. The m6A RNA modification has been found to play a role in this regulation at the RNA level.

Recent and ongoing research is bringing m6A's involvement in various human diseases to light. The development of efficient and high performance m6A detection methods is of importance, as efficient m6A profiling may prove to be a useful diagnostic tool.

The current best m6A detection methods are based on second generation sequencing technologies, where a relatively large amount of analysis steps are required, leaving ample room for the introduction of undesired biases and noise in the analysis.

As the modern sequencing technology continues to improve, sequencing-based analyses and their associated data-analysis tools should not lag behind.

Oxford Nanopore Technologies' MinION sequencing device is portable and has already been used as a diagnostic tool in remote areas (e.g. for rapid strain identification during the ebola crisis (Quick et al., 2016)).

Once an accurate and fast high throughput m6A detection method has been developed, epitranscriptome analysis can be routinely applied for fundamental reseach or medical diagnostic applications.

1.2 Thesis outline

First an overview of the necessary scientific background is provided in Chapters 2-4. Chapter 2 introduces the new biological field of RNA epigenetics, Chapter 3 reviews

sequencing technologies (with a focus on Oxford Nanopore sequencing) and some essential bioinformatics concepts. In Chapter 4, machine learning is discussed, with a focus on deep learning.

Data processing and the development of a predictive model to detect the presence of m6A in RNA transcripts using the basecalled reads originating from an Oxford Nanopore sequencing experiment, is discussed in Chapter 5.

Finally, in Chapter 6, some improvements to the model, as well as some alternative m6A detection approaches are suggested, before ending with some concluding remarks.

CHAPTER 2

RNA EPIGENETICS

2.1 Introduction

While posttranscriptional modifications of RNA were already being described as early as the 1950s, the discovery that an extensive layer of mRNA base modifications influences gene expression is much more recent. In analogy to the field that studies DNA and histone modifications (called epigenetics), the study of RNA modifications and their effects has been termed *RNA epigenetics*.

The amount of different RNA modifications that have been described is currently over 170 and continues to grow (Boccaletto et al., 2018). This exceeds the amount of modifications that are known to decorate DNA by an order of magnitude. The reason for this large discrepancy in amount of alternative nucleotide forms is due to the fact that the main function of DNA is the storage of genetic information, whereas RNA is responsible for translation, as well as a variety of functions of a structural, catalytic or regulatory nature.

The modifications of ribosomal RNA (rRNA) and transfer RNA (tRNA) were the first to be described. The three major types of rRNA methylation are pseudouridines, 2'-O-methyl nucleotides and base methylations (Piekna-Przybylska et al., 2008). Their function generally pertains to quality control in the assembly of ribosomes. Transfer RNA molecules contain more than 10 base modifications on average with base methylations, pseudouridine, 2'-O-methyl nucleotides and modified wobble bases being the most prevalent. Their function is to maintain stability, modulate tRNA folding or to tune the coding capacity of the tRNA molecule (El Yacoubi et al., 2012).

Internal long noncoding RNA (lncRNA) and messenger RNA (mRNA) modifications were discovered later on and consist of N6-methyladenosine (m6A), inosine, N1-methyladenosine (m1A), 5-methylcytidine (m5C), 5-hydroxymethylcytidine (hm5C) and pseudouridine.

In the next section, N6-methyladenosine (m6A), which is the most frequent mRNA modification and the focus of this thesis, will be discussed in more detail.

2.2 N6-methyladenosine

On average, more than three instances of N6-mythyladenosine are present per mRNA molecule in mammalian cells, making it the most abundant mRNA modification in the epitranscriptome. It is most commonly found around stop codons and within long internal exons and is generally associated with the degenerate consensus motif RRACH (R = A or G and H = C, A or U), with exceptions often being attributed to artifacts and false positives. After a thourough search for m6A sites (Dominissini et al., 2013) (the methods used for this are described in Section 2.3), it was concluded that most of the m6A sites are found within expressed gene transcripts and a minority within noncoding genes. Another finding is that a large portion of the m6A sites that were found within the human transcriptome are conserved in the mouse transcriptome (Dominissini et al., 2012), suggesting an important regulatory function of m6A. Gene ontology analysis of differentially methylated transcripts did not yield significant results, which also implies that m6A may serve a more general purpose. N6-methyladenosine has been shown to also be common in prokaryotes, where it usually occurs within open reading frames with a distinct GCCAU consensus motif (Deng et al., 2015).

M6A is currently the only RNA modification for which instances of all three of the signatures of epigenetic regulation (writers, readers and erasers) have been identified in eukaryotes. In this terminology, writers denote the proteins that play a role in the addition of the nucleotide modification. Erasers, on the other hand, are the proteins that remove the modification. Readers are molecules that can bind to the modified nucleotides either directly or indirectly. They are generally associated with their biological functions.

Three writers (Liu et al., 2014) that have been discovered for m6A are methyltransferase like 3 (METTL3), methyltransferase like 14 (METTL14) and Wilm's Tumor Associated Protein (Ping et al., 2014) (WTAP). These three proteins are components of a large protein complex that is responsible for m6A addition. METTL3 and METTL14 are catalytically active components with significant structural similarity, while WTAP is more of an accessory protein that confers selectivity and an appropriate localisation to the complex. WTAP has also been found to be involved in RNA splicing (Horiuchi et al., 2006).

Fat mass and obesity-associated protein (FTO) (Jia et al., 2011) and ALKBH5 (Zheng et al., 2013) are two N6-methyladenosine erasers that belong to the same protein family. Their mechanism of action differs. While FTO achieves demethylation through successive oxidation, with the accumulation of two intermediate oxidation products,

ALKBH5 can directly demethylate m6A without the formation of any detectable intermediates.

It is anticipated that many m6A readers (Wang et al., 2014, 2015) are yet to be discovered, but human antigen R (HuR), human YTH domain family 1 (YTHDF1), human YTH domain family 2 (YTHDF2) and heterogeneous nuclear ribonucleoprotein A2B1 (HNRNPA2B1) are the reader proteins identified to date. HuR or ELAVL1 regulates mRNA stability. The YTHDF proteins contain a conserved domain that directly binds to m6A. YTHDF2 additionally has a functional domain that causes the bound mRNA to be translocated to mRNA decay sites. YTHDF1, on the other hand, promotes mRNA translation efficiency. These opposing YTHDF protein functions may be indicative of a means by which cells can regulate protein synthesis in response to environmental stimuli. HNRNPA2B1 (Alarcón et al., 2015) is the most recently discovered m6A reader and plays a role in alternative splicing. Figure 2.1 illustrates the concept of writers, erasers and readers in the context of m6A.

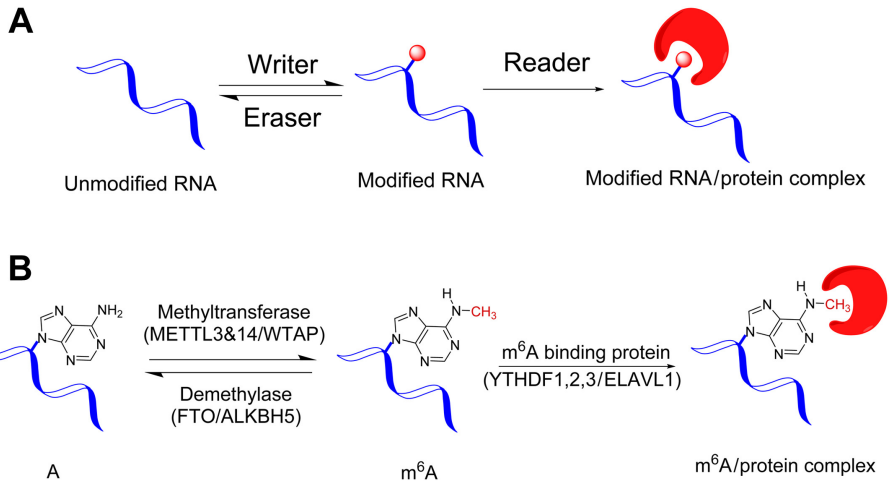


Figure 2.1: **(A)** Signatures of epigenetic regulation. Unmodified RNA is methylated by a writer. Modified RNA can either be converted back to unmodified RNA by an eraser or bound by a reader protein. **(B)** Readers, writers and erasers known for m6A modification (Liu and Pan, 2015).

Various biological processes, including conditions such as prostatitis, several types of cancer, obesity, diabetes, cataract and hepatitis are known to be regulated by, or at least associated with, differences in m6A (Batista, 2017).

2.3 State-of-the-art m6A detection

The earliest detection methods of modified RNA were based on thin-layer chromotography (TLC). These methods are no longer in use except for a recently developed

single-nucleotide resolution method called SCARLET, short for *site-specific cleavage followed by ligation-assisted extraction and thin-layer chromatography* (Liu et al., 2013). High performance liquid chromatography (HPLC) coupled with mass spectrometry (MS) has been used to quantify modified RNA in a sample, but doesn't provide any information on where the modification is localized in the sequence. HPLC-MS also provides a global overview of the amount of modified RNA rather than specific information concerning a single modification.

Reverse transcriptase (RT) can be blocked or paused during primer extension when a modified RNA nucleotide is encountered. This phenomenon allows for the detection of the modifications due to abortive primer extension or nucleotide misincorporation in the vicinity of the modified site. The applicability of these methods for m6A detection has remained limited as it was found to be 'RT-silent'. This means that the presence of m6A in native RNA cannot be detected from aberrant RT behaviour. The use of alternative RT enzymes such as the Thermus thermophilus polymerase (Harcourt et al., 2013) or engineered RT enzymes, however, yield promising results for future applications in m6A detection.

Next generation sequencing-based (NGS) techniques are currently the most widely used m6A detection methods. These techniques generally achieve m6A localization within a sequence either by a (possibly chemically induced) RT signature or antibody-based enrichment. These two strategies can be combined, thus allowing for a decrease in false positives. Immunoprecipitation using m6A-specific antibodies is the current state of the art. M6A-seq (Dominissini et al., 2012) or MeRIP-seq (Meyer et al., 2012) (methylated RNA immunoprecipitation sequencying) split the analysed sample in two parts. In the first part, the RNA is fragmented and the complementary DNA (cDNA) library, resulting from reverse transcription, is sequenced. The second part undergoes the same procedure after enrichment for modified RNA through immunoprecipitation. M6A sites are then identified as regions where the amount of reads is increased upon enrichment. A peak-calling algorithm such as exomePeak (Meng et al., 2014) can efficiently identify these regions. Figure 2.2 displays an overview of the m6A-seq protocol.

Mapping m6A at Individual-Nucleotide Resolution Using Crosslinking and Immunoprecipitation (MiCLIP) (Grozhik et al., 2017) is a single-nucleotide resolution method in which anti-m6A antibodies are UV-crosslinked to the RNA. Specific mutational signatures arise after reverse transcription of crosslinked RNA, from which the m6A position can be deduced.

Promising new developments in m6A detection are the application of single molecule sequencing technologies (discussed in Section 3.2.4) that allow the modified RNA

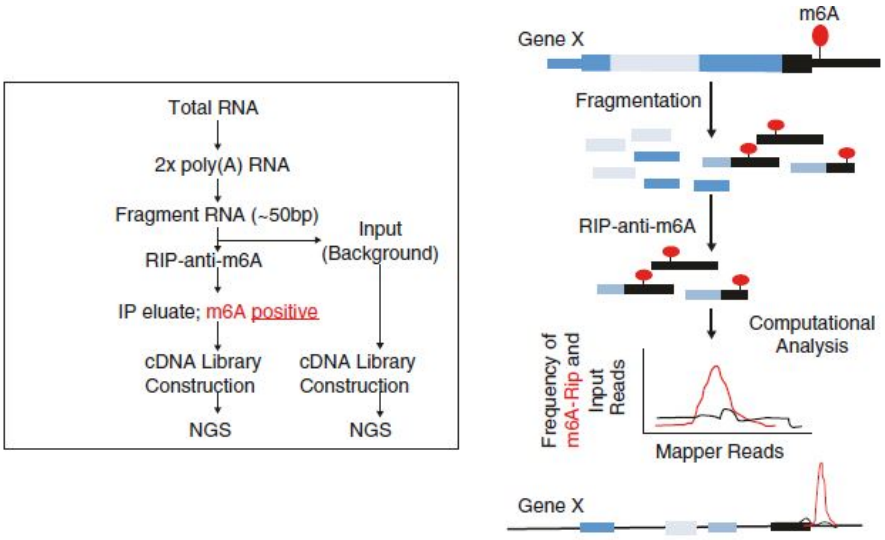


Figure 2.2: Work flow (left) and schematic diagram (right) of the m6A-seq protocol (Lusser, 2017). RNA immunoprecipitation (RIP) is carried out on fragmented poly(A) RNA using anti-m6A antibodies followed by RNA sequencing of the enriched m6A positive RNA fragments with a background control of non-immunoprecipitated RNA fragments.

bases such as m6A to be detected directly without the need for intermediate reverse transcription (as was the case with m6A-seq). Pacific Biosciences' Single Molecule Real Time (SMRT) sequencing platform has already been shown to be capable of detecting m6A residues (Vilfan et al., 2013). Oxford Nanopore Technologies (ONT) has expanded its single molecule sequencing technology to allow direct RNA sequencing, this should make it possible to develop tools for the detection of m6A, as these are currently lacking.

2.4 The MethylTranscriptome Database

Hui Liu and colleagues assembled a database containing publicly available MeRIP-seq data. This database is called the MethylTranscriptome Database (MetDB) (Liu and Pan, 2015; Liu et al., 2018) and consists of more than four hundred thousand m6A sites from over 185 samples originating from seven different species and twenty-six independent studies.

After quality control, trimming and alignment (these steps will be elaborated in the next chapter), peak calling was performed. Each predicted m6A peak constitutes a new data instance in the database. Positional information, quality scores (p-values and false discovery rate scores) as well as the source of the data are reported for each predicted m6A peak in the database.

While the resolution of m6A-seq is limited to about 100-nucleotide-long regions of transcripts, a single-base m6A dataset was obtained by attributing the m6A peaks to the nearest instance of the degenerate sequence motif (RRACH). The distance of the called peak to the nearest motif is provided and can be used as a confidence score. Other quality scores are not reported in the single-base m6A dataset. Figures 2.3 and 2.4 depict the distribution of these scores and the sequence logo of the sequences in the database, respectively.

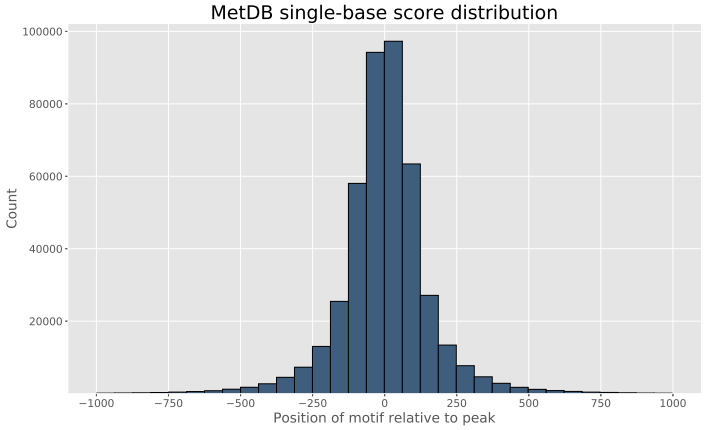


Figure 2.3: Histogram of the scores, which are the relative positions of the called peaks with respect to the nearest RRACH motifs. About 90% of the entries have a motif within a distance of 200 nucleotides from the m6A peaks.

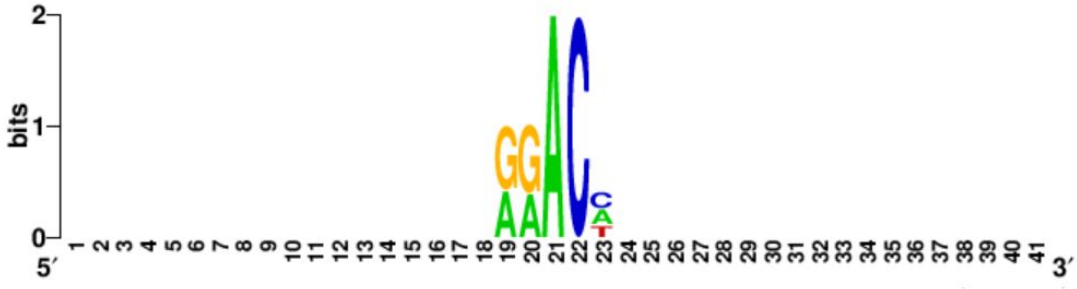


Figure 2.4: Sequence logo of the MetDB entries ranging from 20 nucleotides upstream to 20 nucleotides downstream of the m6A sites. A sequence logo (Schneider and Stephens, 1990) is a visual portrayal of the nucleotides that are conserved in a collection of sequences. The vertical axis (i.e. the height of the letters) represents the information content of the associated position, measured in bits. The RRACH motif (where R is A or G and H is C, A or T) is clearly present in all of the entries of the MetDB single base database.

CHAPTER 3

BIOINFORMATICS AND SEQUENCING

3.1 Introduction

Technological progress in the last few decades gave rise to the generation of an immense amount of biological data. The diverse field of bioinformatics develops methods to gain insight in and value from this data. The subfields of genomics and proteomics develop and use a wide range of tools to analyse nucleic acids and proteins. The fast-paced development of nucleic acid sequencing technologies is a significant contributing factor of the recent surge in biological data. These sequencing technologies are discussed next.

3.2 Sequencing technologies

3.2.1 Sanger sequencing

Nucleic acid sequencing (mainly in the form of DNA sequencing) has been around since 1977 when Frederick Sanger and colleagues developed the first method (Sanger et al., 1977). Sanger sequencing is based on the selective incorporation of dideoxynucleotides (which cannot be elongated) by DNA polymerase during in vitro DNA replication. The DNA sequence can be inferred from the resulting pattern after polyacrylamide gel electrophoresis. Although Sanger sequencing is still being used today for small-scale experiments, its limited throughput and read length (although the reads are still long compared to some of the more recent methods) gave way to the emergence of next generation sequencing (NGS) technology.

3.2.2 Next generation sequencing

Next generation sequencing technologies are characterised by their tremendously increased throughput and concomitant decrease in cost per base. These technologies also require DNA amplification as part of their library preparation. When RNA molecules are to be sequenced (known as RNA-seq), an intermediate reverse transcription step is required to obtain a complementary DNA (cDNA) library before sequencing. Several different competing technologies were developed, each having their strengths and weaknesses. Among these technologies, illumina sequencing gained the upper hand due to its large throughput (Levy and Myers, 2016).

3.2.3 Illumina

What follows is a general and brief description of the Illumina sequencing principle. The protocol of this sequencing platform consists of four steps: sample preparation, cluster generation, sequencing and data analysis. During sample preparation, adaptors are added to the DNA fragments. One of the functions of these adaptors is to allow hybridisation to oligonucleotides on the flow cell. During cluster generation, DNA strands complementary to the sample DNA are synthesized. Afterwards, the double stranded DNA bound to the flow cell is denatured, and the original template is washed away. The unbound adaptor sequence of the single stranded molecule subsequently hybridizes with a nearby oligonucleotide on the flow cell and bridge polymerase chain reaction (PCR) (Mullis et al., 1986) follows. This bridge PCR is repeated many times and the reverse strands are removed so that only forward strands remain. The next step is massively parallel sequencing by synthesis on the flow cell, where DNA polymerase catalyses the incorporation of fluorescently labeled deoxyribonucleotide triphosphates (dNTPs) into a DNA template strand during sequential cycles of DNA synthesis. The nucleotides are identified by nucleotide-specific fluorophore exitation. In paired-end sequencing, the reverse strands are also sequenced in order to increase the quality of sequence alignment.

3.2.4 Third generation sequencing

Third generation sequencing, or single molecule sequencing, is a promising recent development in sequencing technology. The limited read length of the first two sequencing generations, as well as the requirement for an amplification step that can introduce bias, are two incentives for the development of long-read technologies.

Pacific Biosciences and Oxford Nanopore Technologies are two companies that each have successfully developed single molecule sequencing technologies.

Pacbio SMRT sequencing

Pacific biosciences' single molecule real-time (SMRT) sequencing platform (Zhu and Craighead, 2012) achieves single molecule sequencing by employing DNA polymerase molecules in nanoscale observation chambers called Zero Mode Waveguides (ZMW), in which the incorporation of fluorescently labled nucleotides is recorded. The sample DNA is first circularised by ligation of hairpin adaptors on both sides. Due to the circularity of the resulting structure (called a SMRTbell), the insert sequence can be covered multiple times. Therefore, an accurate consensus sequence, called a circular consensus sequence, can be determined.

Oxford nanopore sequencing

Another single-molecule sequencing platform is Oxford Nanopore sequencing. This technology features prominently in this thesis. The basic principle of nanopore sequencing is the nucleotide-specific disruption of ionic current measured as single strands of nucleic acids are ratcheted through a nanopore protein embedded in a synthetic polymer membrane with high electrical resistance. Several protein nanopores have been researched and employed by Oxford Nanopore Technologies (ONT), including the alpha-hemolysin pore protein obtained from Staphylococcus aureus. In the case of native DNA sequencing, a choice must be made between single pass 1D and double pass 2D reads, which represents a trade-off between accuracy (2D) and throughput (1D). In 2D reads, the two DNA strands are ligated by a hairpin adaptor and they are sequenced consecutively. 1D reads involve the analysis of a single strand of the nucleic acid molecules, which generates data of slightly lower quality albeit at a higher sequencing rate.

The library preparation is rather simple and consists of ligation of a leading and a trailing adaptor to the 5'- and 3'-ends respectively. These adaptors allow the addition of a motor protein at the 5'-end, aiding in the concentration of the nucleic acids near the nanopore. The motor protein unwinds double stranded DNA, thereby allowing single-file passage of the nucleotides through the nanopore. In the case of 2D sequencing, a hairpin loop connects the template strand to the complementary strand so that these are sequenced in tandem, and a concensus sequence with improved accuracy can be attained. Although some authors claim that SMRT sequencing is currently a more matured sequencing platform (Ardui et al., 2018) as it delivers a better data quality

overall (Weirather et al., 2017), ONT maintains a competitive edge due to the lack of expensive (fluorescently labeled) reagents and polymerase enzymes, as well as the convenient portability and the high speed operation with an intriguing 'run untill' option that gathers data untill the user is satisfied. The chemistry (protein and reagents) and data anaysis workflow are also frequently updated to increase performance.

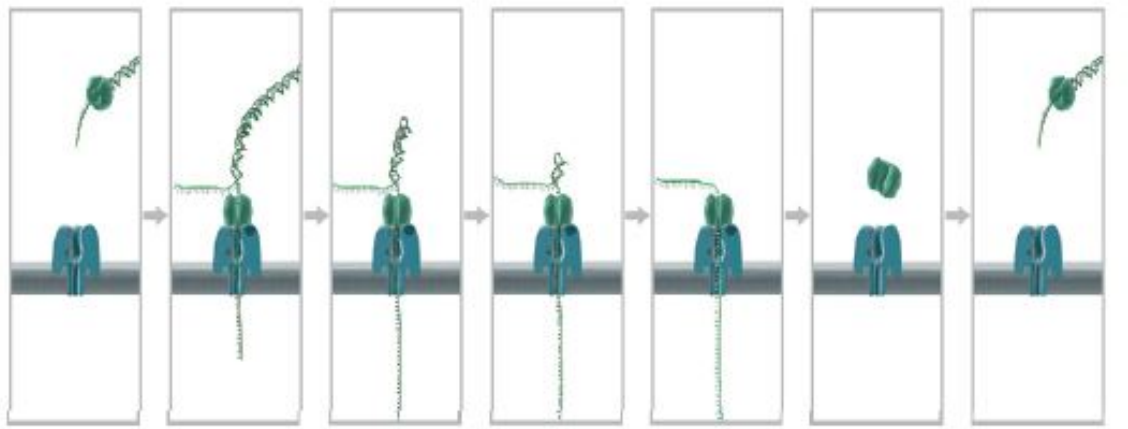


Figure 3.1: An illustration of the oxford nanopore sequencing process. When the DNA-enzyme complex approaches the nanopore , the single stranded DNA is pulled through the aperture. For 2D reads, the reverse strand is connected to the forward strand by a hairpin adaptor. It is pulled through the nanopore after the forward strand, allowing for a higher quality consensus sequence to be achieved.

RNA sequencing was already possible through cDNA after reverse transcription, but the recent development of direct RNA sequencing (Garalde et al., 2018) grants us access to the RNA molecule in its native state. Before this development, nearly all RNA sequencing endeavours relied on reverse transcription, making discoveries that are inhibited by this intermediate step impossible. The ability to infer the presence of modified RNA residues with single-nucleotide resolution directly from the native RNA sequence should be possible. This has already been attempted with synthetically modified RNA, with moderate success (Liu et al., 2019). An overview of the straightforward library preparation for direct RNA sequencing is displayed in Figure 3.2.

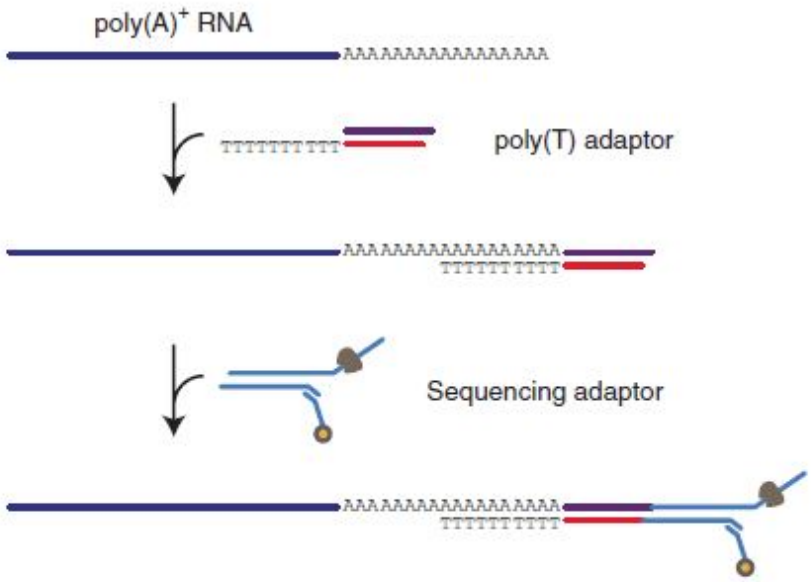


Figure 3.2: The simple library preparation steps for direct RNA sequencing (Garalde et al., 2018). Using T4 DNA ligase, the poly(A)+ RNA in the sample is ligated to a poly(T) adaptor with a double stranded portion at its end. Sequencing adaptors preloaded with a tether and a motor protein are then ligated onto the overhang of the previous adaptor by T4 DNA ligase.

3.3 Basecalling

An important and non-trivial part of the sequencing process is the acquisition of a nucleotide sequence starting from the raw output of a sequencing machine. This step is called basecalling and is generally integrated in the proprietary technology of the respective sequencing platforms. In several cases, third party basecalling softwares have been developed for their open source nature, increased performance or other specific purposes. The output of a basecalling algorithm is typically stored using a fastA or fastQ file format (Cock et al., 2010). The former contains a header/identifier followed by the basecalled sequence, while the latter additionally contains quality scores. The quality scores in FASTQ files are called Phred scores (Q), calculated using the following equation:

$$Q = -10 \log_{10}(P) \tag{3.1}$$

where P is the estimated probability of incorporating a wrong base.

In the case of oxford nanopore sequencing, the raw signal (electric current) output, which is stored in Fast5 files (a variant of HDF5 files), is segmented in events. A spe-

cialized algorithm is used to predict the bases that correspond to the events. Initially hidden markov models (HMMs) were used (Bishop, 2006). Current state-of-the-art ONT basecallers are bidirectional recurrent neural networks (RNN), such as ONT's Albacore and the open source DeepNano (Boža et al., 2017). Recurrent neural networks are discussed in Section 4.3.3.

3.4 Sequence alignment and quality control

After acquiring the sequence from basecalling, its location of origin in the genome can be elucidated by so-called alignment algorithms. Prior to sequence alignment, quality control of the sequence should be performed (e.g. by fastQC), possibly followed by 'trimming off' parts of the sequence (e.g. residual adaptor sequences) that don't meet the imposed quality requirements (e.g. using Trimmomatic (Bolger et al., 2014)).

A variety of alignment algorithms have been described, with many being designed to address specific problem settings. Other than the sequence to be aligned, a genome assembly and optionally a Gene Transfer Format (GTF) annotation file, containing information concerning the location of genes, transcripts and exons, should be provided to the alignment algorithm. The output is a Sequence Alignment Map (SAM) file. SAM files contain several characteristics of the aligned reads. Three interesting reported characteristics are the genomic coordinates (chromosome and position), the mapping quality, and the CIGAR string. The latter is an abbreviation of Concise Idiosyncratic Gapped Alignment Report and provides a detailed yet compact description of the differences and similarities between the reads and the reference sequence to which they are aligned. It is a string of letters and numbers, in which the letters denote operations and the numbers are the amount of consecutive bases to which the operations apply. The most frequent CIGAR string operations are displayed in Table 3.1.

Table 3.1: Common CIGAR string operations.

Operation	**Description**
M	Match : Base position in reference corresponds to base in read.
I	Insertion : Base abscent in reference but present in read.
D	Deletion : Base abscent in read but present in reference.
N	Skip : Range of bases not present in read.
S	Soft clipping : Non-aligned bases at the read ends present in read.
H	Hard clipping : Non-aligned bases at the read ends removed from read.

An alignment tool developed specifically for ultra-long reads such as those resulting from an Oxford Nanopore direct RNA sequencing experiment, is Minimap2 (Li, 2018).

CHAPTER 4

MACHINE LEARNING

4.1 Introduction

According to former Baidu Chief Scientist Andrew Ng, *artificial intelligence (AI) is the new electricity*. In this analogy, the impact of AI on our lives is compared to that of the replacement of steam powered machines with those using electricity at the end of the 19th century. Currently, many businesses and domains of industry are being transformed by AI, more specifically by machine learning. Machine learning is a subfield of artificial intelligence that focuses on automatically finding patterns in data without being explicitly programmed. Figure 4.1 illustrates a typical data analysis workflow in which machine learning constitutes the final steps.

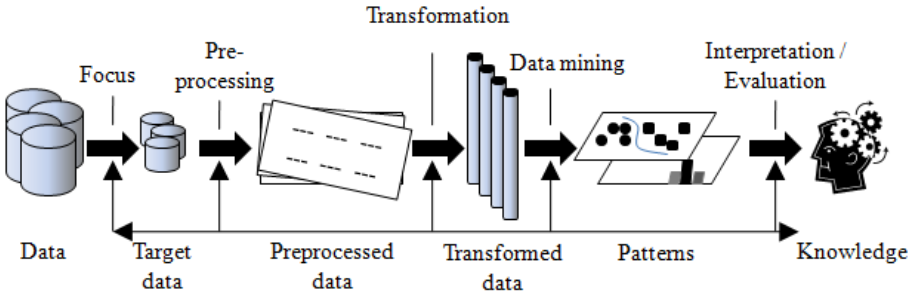


Figure 4.1: Knowledge Discovery from Databases (Fayyad et al., 1996). After the necessary available data is selected, it is preprocessed and transformed into a structure that lends itself to the appropriate data mining (machine learning) analysis that follows. This machine learning step can consist of one or more of a variety of methods that will be discussed in the next section. Finally the output is interpreted and/or evaluated and can increase our domain knowledge.

4.2 Basics

4.2.1 Types of machine learning

The three main branches of machine learning algorithms are unsupervised, supervised and reinforcement learning. The latter is based upon a learning algorithm that

learns to take appropriate actions in a specified situation in order to maximize a reward. A difficulty in reinforcement learning is that rewards are often only amassed after a sequence of actions made by the learning algorithm in its environment. A popular application of reinforcement learning is a software agent learning to play games. AlphaGo (Silver et al., 2017) is a program that beats any human player in the Chinese board game Go.

Unsupervised learning is concerned with finding relationships between variables without an outcome or response variable having to be predicted. These methods are used to analyse unlabelled input features. They can be used to attempt to cluster observations into distinct groups in order to ascertain similarities between observations, or to perform density estimation. An example is news websites that cluster together articles with related content, providing the reader with as much relevant information as possible. Another example is the clustering of cancer cell types based on their gene expression profiles. This is done to provide insights in viable treatment strategies. Unsupervised methods are often applied in exploratory data analysis.

In supervised learning, the goal is to predict an outcome variable (that is often impractical to measure) using a set of input variables. A wide range of methods have been developed for this purpose. The importance of having a variety of methods at our disposal is exemplified by David Wolpert's "no free lunch" theorem, which states that no single machine learning algorithmn outperforms any other when their performance is averaged over all possible problems (Wolpert and Macready, 1997).

Depending on the nature of the response variable, a distinction is made between regression problems, for which a continuous output value is to be predicted, and classification problems, for which the output is constrained to a discrete set of classes.

Another frequently made distinction in supervised learning methods is whether the the model is parametric. Parametric models, such as linear regression, make an assumption about the functional form of the model (e.g. the response is linear). This effectively reduces the modelling effort to the estimation of parameter values that maximize the prediction accuracy on the training data (which is the data used to fit the model, consisting of both the input variables and their associated true response value). Non-parametric models, on the other hand, do not require any assumptions about the functional form of the outcome, but merely seek to deliver predictions that are as similar as possible to the training data's response values for similar input values. A simple but often effective example is the K-nearest neighbours algorithm (KNN) that assigns the most prevalent outcome class (C) among the K training data

instances with input values closest to those of the test point to be predicted.

$$y_i = Argmax_C\Big(\frac{1}{K}\sum_{j\in\mathcal{N}_i} I(y_j = C)\Big) \tag{4.1}$$

In the above formula, $\mathcal{N}_i$ represents the K training data instances, who's feature vectors x_j lie closest to the feature vector x_i to be classified, according to an appropriately chosen distance metric (e.g. Euclidian distance). The value y, is the class associated with an input feature vector x. $I(y_j = C)$ is the indicator function which returns 1 if $y_j = C$ and 0 otherwise. The advantage of non-parametric models is that they often perform well on data of relatively low dimensionality, without requiring any prior knowledge about the true functional form or assumptions that limit the flexibility of the models. A drawback of non-parametric models is that quite a lot of data is required to sufficiently cover the input space (Marimont and Shapiro, 1979). Another drawback is that the training data is required when doing predictions with non-parametric models. When using parametric models, the training data can be discarded once the optimal parameter values have been acquired (see Section 4.2.2).

4.2.2 Parameter optimisation

First a distinction should be made between parameters and hyperparameters. Parameters are variables that are part of parametric machine learning models and can be adjusted during training. They are the part of the model that is learned from the training data. Hyperparameters are not estimated from the data, but are manually specified before the model is trained. An example of a hyperparameter is K in the K nearest neighbours algorithm.

A loss function quantifies our dissatisfaction with the predictions of a model for a given set of values of its parameters. In order to achieve the best performance of the model, the loss function should be minimized.

In some machine learning algorithms such as linear regression, a closed form solution exists to the problem of finding the optimal parameter values (i.e. the method of least squares). In other cases, iterative convex or non-convex optimisation is required to acquire the most suitable parameter values that maximise the model's performance. Gradient descent is such an iterative method that attains parameter values corresponding to a (local) minimum in the loss function by taking successive steps in the direction that locally leads to the greatest decrease in the loss function. This direction corresponds to the negative gradient of the loss function, evaluated at the current

parameter values of the model.

$$w_k = w_k - \alpha \frac{1}{N} \sum_{i=1}^{N} \frac{\delta \mathcal{L}(y_i, \hat{y}_i)}{\delta w_k} \tag{4.2}$$

In Equation 4.2, a gradient descent step to update the parameter w_k is displayed. The learning rate is denoted by α, N is the amount of training observations and $\mathcal{L}$ is the loss function applied to the prediction $\hat{y}_i$ and the true label y_i.

In order to tune the hyperparameter values in a less manual fashion, resampling methods such as cross-validation are commonly used. Cross-validation involves repeatedly excluding a different specified amount of observations from the training procedure and evaluating the mean performance of the resulting models on the excluded observations. These hyperparameter tuning methods will not be discussed in more detail here, as it is not standard to apply them to the methods used in this thesis.

4.2.3 Model capacity and regularisation

The used machine learning method, the amount of parameters included, as well as the degree of optimisation (e.g. the amount of passes through the training data in the case of neural networks, see Section 4.3.2), are choices that influence the complexity of the fit. Some methods are more flexible than others, meaning that they can more easily attain the structure and patterns in the training data. This degree of flexibility is termed the model capacity. There are, however, patterns present in the training data that are not inherent to the process being modelled, but rather to random noise in the data. It is undesirable to include this random noise in the model, as it doesn't describe the process being modelled, resulting in a decrease in predictive performance when classifying unseen data. This problem of *overfitting* the model to training data can be mitigated by either choosing a simpler model or by increasing the amount of data. However, when an overly simple model is chosen, certain patterns in the data that require a certain degree of model flexibility can no longer be modelled adequately. This trade-off between underfitting models that make invalid simplifying assumptions and overfitting models that model irrelevant noise in the training data is known as the bias-variance trade-off (see Figure 4.2). Both ends of the trade-off bring about an unsatisfactory performance of the model when tested on new data.

A common way to reduce overfitting is with regularisation. Any modification to a learning algorithm that results in a decrease of the generalisation error (i.e. the error made on unseen test data), but not of the training error, is known as regularisation. Various methods exist to penalize model complexity through regularisation. A popular

regularisation method is to penalize the loss function with a parameter norm function. In many models (e.g. polynomial regression), parameter values tend to become large as the model starts overfitting to the training data. This can be seen in Figure 4.2. Thus, including a measure of the magnitude of the parameters in the loss function can counteract overfitting. Two such measures that are widely used are the L1 and L2 norms. The L1 norm, also known as the lasso in linear regression (Tibshirani, 1996), is given by:

$$\sum_{i=1}^{n}\left(y_i - w_0 - \sum_{j=1}^{p} w_j x_{ij}\right)^2 + \lambda \sum_{j=1}^{p} |w_j| \tag{4.3}$$

The L2 norm, also known as ridge regression (Hoerl and Kennard, 1970) or weight decay, is given by:

$$\sum_{i=1}^{n}\left(y_i - w_0 - \sum_{j=1}^{p} w_j x_{ij}\right)^2 + \lambda \sum_{j=1}^{p} w_j^2 \tag{4.4}$$

In Equations 4.3 and 4.4, the first term is the residual sum of squares, while the second term is the norm penalty. The amount of observations in the training set is denoted by n, and p is the total number of features included in the model. The jth input feature of the ith observation, is denoted by x_{ij} .The parameter λ is a hyperparameter that can be selected using cross-validation (see Section 4.2.2). Both of the described parameter norms are useful in lowering the generalisation error by decreasing the model's variance. An advantage of the L1 norm is that it performs model selection by setting the coefficients of unnecessary terms to zero.

4.2.4 Model performance evaluation

In order to assess the value of a model it must be tested with a separate dataset from the dataset that was used to optimise the parameters. Otherwise the performance would be overestimated due to the training-set specific noise that would be predicted. Ideally a separate validation set is included as well for use during hyperparameter tuning and model selection.

Simply reporting the error rate

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i) \tag{4.5}$$

of a classifier (where $\hat{y}_i$ is the predicted class, y_i is the true class of observation i, and n is the amount of observations) to evaluate its performance is often not satisfactory because it does not take difficulties arising from class imbalances into account. An example is a binary classification problem in which 99% of the observations belong to the first class. if the classifier were to assign every observation to the first class, it would have an error rate of only 1% despite being useless. An often used performance

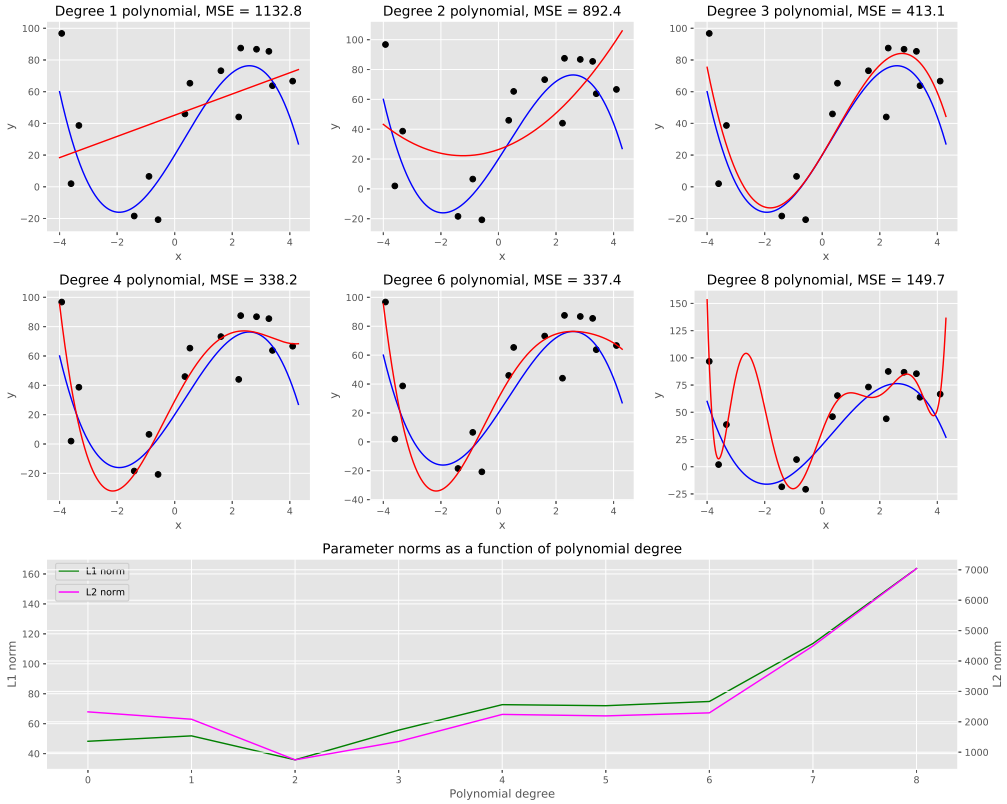


Figure 4.2: **Bias-variance trade-off illustration**. A data generating function $f(x) = -2x^3 + 2x^2 + 30x + 20$ with $x \in [-4, 4]$ (shown in blue) was used to generate 15 data points, subject to gaussian noise $\epsilon \sim \mathcal{N}(0, 20)$. Polynomial functions of degree 1, 2, 3, 4, 6 and 8 (shown in red) were respectively fitted to the generated sample in order to see the bias-variance trade-off in action. Upon visual inspection, the third degree polynomial fits the data generating formula best. This is expected, as the ground truth is a polynomial of degree 3. As the degree of the polynomial increases, the training data points can be fitted more accurately. A straight line and a parabola are too simple models, they are incapable of fitting the training data and suffer from bias. As the degree of the polynomial becomes increasingly larger than three, the model 'tries too hard' to fit the data points and starts fitting the sample-specific noise. The resulting polynomials would look different when a new noisy sample is drawn from the data generating function. The degree 8 polynomial has high variance and drastically overfits the training sample. As the model overfits and attempts to precisely fit the sample data, it displays more erratic behavior and the coefficients take on large values. This can be observed in the increasing parameter norms ($L1 = \sum_{j=1}^{p} |w_j|$ and $L2 = \sum_{j=1}^{p} w_j^2$, see Equations 4.3 and 4.4) at the bottom of the figure. The reported mean squared error (MSE) is a measure of how badly the model fits to the sample data and always decreases as the degree of the polynomial increases. A polynomial of degree n, with n the sample size, is capable of precisely passing through every single point in the training sample (MSE = 0), but drastically overfits and would be rather useless on a new sample. It is clear that regularisation can be achieved by penalizing the MSE (which decreases with complexity) with a parameter norm penalty (which increases as the model overfits).

measure that doesn't fall victim to the above phenomenon is the ROC AUC (Area Under the Curve) score. To describe this performance measure, some concepts such as confusion matrix, sensitivity, specificity and the receiver-operator characteristic curve (ROC) should be introduced. For a classifier with C classes, the *confusion matrix* is a C by C matrix in which any element (i, j) is given by the amount of observations classified in class i while actually belonging to class j. Elements on the diagonal correspond to correct classifications. In the case of binary classification, the confusion matrix is a 2x2 matrix. Terminology associated with these elements is given by Figure 4.3. The *Sensitivity*, *true positive rate (TPR)* or *Recall* of a model is the ratio of the

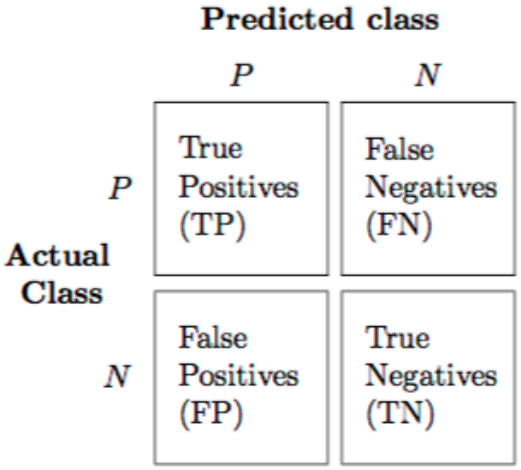


Figure 4.3: Terminology for the confusion matrix in case of binary classification (McElwee, 2018). The column labels are the classes predicted by the binary classification model, whereas the row labels denote the true classes to which data belong.

amount of correctly predicted positives to the total amount of positive observations,

$$Sensitivity = TPR = \frac{TP}{TP + FN} \tag{4.6}$$

whereas the *specificity* is the ratio of the number of correctly predicted negatives to the total amount of negative observations.

$$Specificity = \frac{TN}{TN + FP} \tag{4.7}$$

When the threshold of the minimum output probability to classify an observation as a positive is varied, the values in the confusion matrix change as well as the sensitivity and specificity values. A plot of the sensitivity versus the *false positive rate (FPR)*, which is equal to one minus the specificity, is a *ROC curve* and the area under this curve (AUC) gives a measure of the model's overall performance.

$$1 - Specificity = FPR = \frac{FP}{TN + FP} \tag{4.8}$$

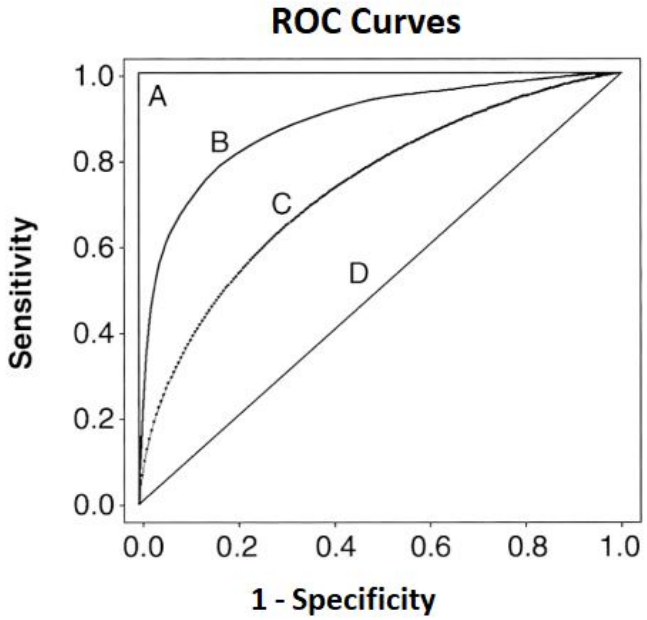


Figure 4.4: ROC curves of four models of decreasing performance denoted by the letters A to D. Model A is a perfect model with an AUC score of 1. Model D, with an AUC score of 0.5, performs no better than a random coin toss.

Alternative performance metrics for the evaluation of binary classification models, are precision and recall. The *precision* of a model is the fraction of predicted positives that are true positives.

$$Precision = \frac{TP}{TP + FP} \tag{4.9}$$

A precision-recall curve is a plot of the precision of a model versus its recall for different thresholds of the output probabilities.

4.3 Deep learning

4.3.1 Motivation, advantages and drawbacks

Neural network models have achieved groundbreaking results and are the state of the art for certain problem settings such as image recognition (Huang et al., 2016), natural language processing (Vaswani et al., 2017) and speech recognition (Chiu et al., 2017). A general trend in machine learning methods is the observed trade-off between model complexity (flexibility) and interpretability. Regularised linear regression models and deep neural networks are two extremes of this spectrum. It has

been shown that a sufficiently large neural network model is capable of representing any abitrarily complex function (Hornik et al., 1989). The interpretability of the outcome, however, is currently still lacking. On the other hand, linear regression is more straightforward to interpret, but only performs well when the data to be fitted is of a linear nature. Neural network models are harder to train compared to many classic methods due to the trial-and-error procedure of tuning a relatively high amount of hyperparameters. Furthermore, the presence of local minima in the non-convex loss function hinders parameter optimisation. A property of Neural networks is that they often require a rather large amount of data to achieve a decent performance, which is inconvenient if the data cannot be easily and cheaply gathered. This "data hunger" of neural networks can also be seen as a favorable property, because the models have the capacity to continue to increase their performance with increasing data, while many more traditional methods will have reached a plateau of performance.

An important advantage of deep learning is that feature engineering is not needed. Feature engineering is often a challenge in other machine learning methods, but less so in deep learning, as an appropriate neural network architecture should be capable of learning its own features from the input data through a series of new representations that correspond to the successive layers.

4.3.2 Neural network model components

Input units, hidden units and output units

The first layer of a neural network is the input layer, which contains one unit per variable value of the observations. After the input layer there are one or more hidden layers, each containing one or more hidden units. The structure of a single hidden unit is displayed in Figure 4.5. Activation functions introduce non-linearities, allowing the artificial neural networks (ANNs) to map non-linear correlations. It is a requirement that activation functions are non-linear, otherwise the resulting neural network would also turn out to be a linear model, incapable of learning non-linearities in the data.

Common activation functions are the Sigmoid, Tanh (hyperbolic tangent) and ReLU (rectified linear unit) functions. Sigmoid units are essentially logistic regression units that map the linearly combined inputs to a value between 0 and 1.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4.10}$$

In Equation 4.10, z is a linear combination of the p input variables, given by $\sum_0^p w_i x_i + b$. Here b is a bias term. Tanh units, which are actually scaled sigmoid functions,

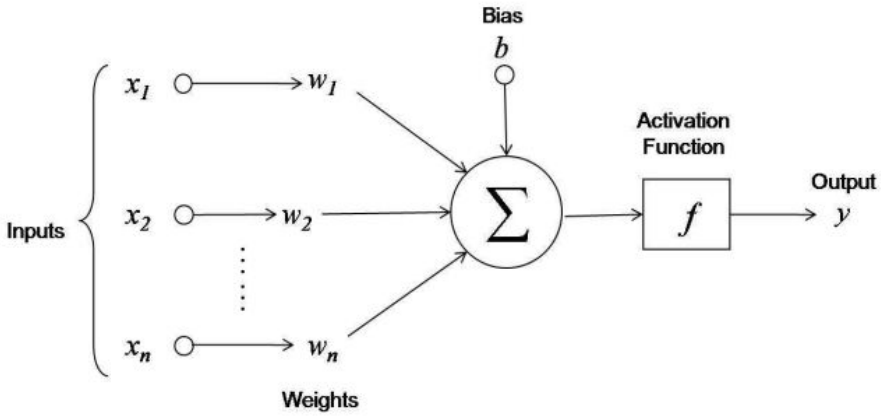


Figure 4.5: Computational graph of a single hidden unit (or neuron) of a neural network. The inputs $\{x_1, x_2, x_3, ..., x_n\}$ are linearly combined with weights $\{w_1, w_2, w_3, ..., w_n\}$, and a bias term b is added. A non-linear activation function f is then applied to the linearly combined inputs to produce the output y of the neuron.

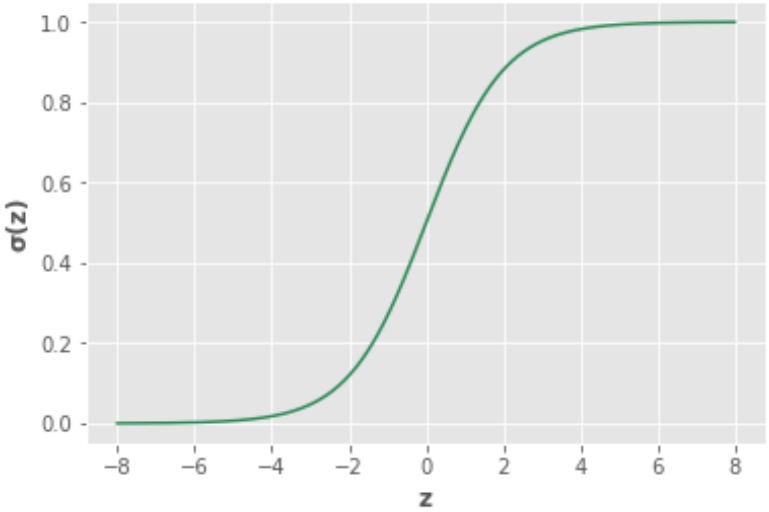


Figure 4.6: Sigmoid activation function.

behave similarly but map to a value between -1 and 1.

$$\tanh(z) = 2\sigma(2z) - 1 \tag{4.11}$$

A major disadvantage of using sigmoid or tanh units is that they suffer from vanishing gradients in deep neural networks. In this phenomenon, learning is inhibited by the small values of the gradients near the asymptotes of the activation function. Another factor contributing to vanishing gradients is the fact that the derivative of the sigmoid function is always smaller than 1, which means that some of the gradients obtained through the chain rule of calculus (more specifically backpropagation, which is explained below) contain many multiplied values smaller than 1. The ReLU activation

function gives an output of z if z is positive and 0 otherwise. It has several advantages and is often the first choice when assembling an ANN model. The advantages of ReLU activations include that they are less susceptible to vanishing gradients than sigmoid units, they are computationally inexpensive and allow a network to obtain beneficial sparse representations (Glorot et al., 2011).

$$ReLU = \begin{cases} z, & if\ z > 0 \\ 0, & Otherwise \end{cases} \tag{4.12}$$

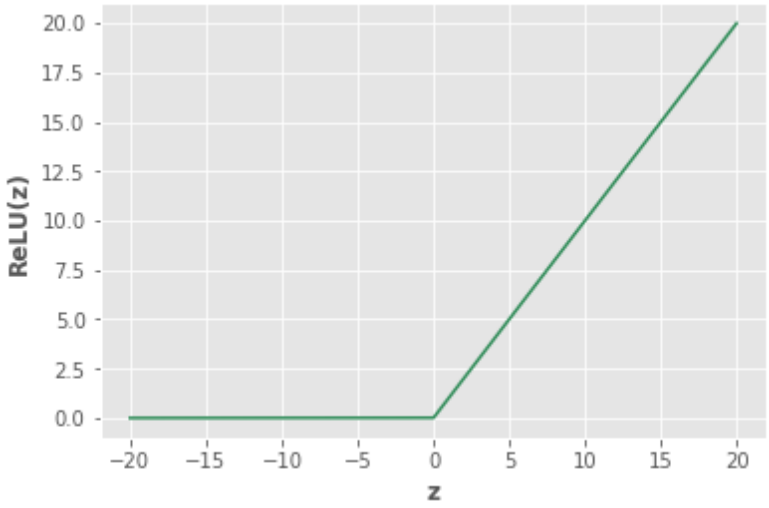


Figure 4.7: Rectified Linear Unit (ReLU) activation function.

The outputs of the last hidden layer are transformed into the model's prediction values in the output layer. For classification problems, the softmax output function is typically used. A softmax function transforms the output of the last hidden layer to K values that are positive and sum to one, where K is the amount of classes. The softmax output values are essentially class probabilities. The loss function is applied to the output layer.

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}} \tag{4.13}$$

The exponentiation of the values in the softmax formula makes sure that all values are positive, while the normalisation in the denominator causes the outputs to sum to one so that they can be interpreted as class probabilities. It can be shown that softmax with $k = 2$ is equivalent to logistic regression. The weights associated with the units of neural networks are initialized randomly or according to a certain initialization strategy (Glorot and Bengio, 2010) , but may never all be zero, as this would hamper the training process.

Backpropagation and optimisation

The gradients needed to adjust the parameters of the ANN during training are obtained by a method called backpropagation (Rumelhart et al., 1986). This algorithm is based on the chain rule of calculus, used to iteratively compute gradients w.r.t. the output of the loss function along the nodes of a computational graph of the model. The strength of backpropagation is that it utilises an efficient order of operations and optimally stores the values of sub-expressions that are used more than once. The resulting computed gradients of the parameters with respect to the loss function are used by the optimisation method to adapt the parameter values.

Common gradient descent is not advisable when training neural networks, because a vast amount of computation would be needed to take a single step along the negative gradient. Stochastic gradient descent (SGD) is more appropriate. In SGD, gradient steps are taken w.r.t. a subfunction of the loss function based on a sampled limited amount of observations called minibatches. This significantly speeds up the learning process.

$$w_k = w_k - \frac{\alpha}{m} \sum_{i=1}^{m} \frac{\delta \mathcal{L}(y, \hat{y}_i)}{\delta w_k} \tag{4.14}$$

Equation 4.14 is a single SGD step to adjust a parameter w_k of a model that currently predicts an outcome $\hat{y}_i$ on an example x_i. Herein α denotes the learning rate, m is the size of the minibatches and $\mathcal{L}$ is the loss function. Several improvements have been made to SGD (Goodfellow et al., 2016), of which adam (adaptive momentum) (Kingma and Ba, 2014) is most widely used.

Regularisation in neural networks

Various regularisation strategies are employed by neural network models including the norm penalties to the loss function discussed in Section 4.2.3. Two important neural network regularisation methods, batch normalisation (Ioffe and Szegedy, 2015) and dropout (Srivastava et al., 2014), are often found in a deep learning practitioner's toolbox.

Batch normalisation applies the following transformation after linear combination of the outputs of the previous layer but before the current layer's activation function:

$$BN_{\gamma,\beta}(x_i) = \gamma\Big(\frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}\Big) + \beta \tag{4.15}$$

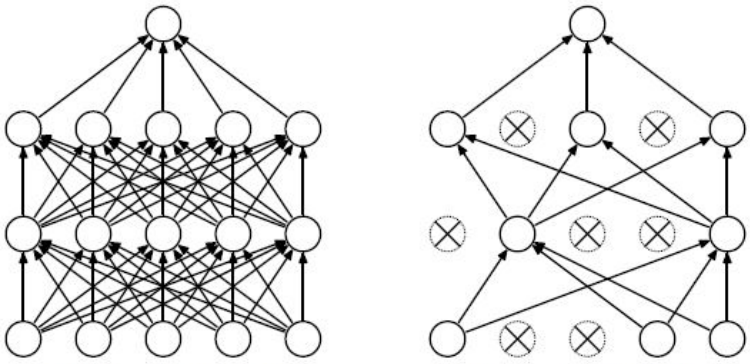
In Equation 4.15, $\mu_{\mathcal{B}} = \frac{1}{m}\sum_1^m x_i$ is the mean estimate calculated on the minibatch $\mathcal{B} = \{1, \ldots, m\}$ and $\sigma^2_{\mathcal{B}} = \frac{1}{m}\sum_1^m (x_i - \mu^2_{\mathcal{B}})$ is the biased variance estimate calculated on

the same minibatch. A small constant ϵ is included in the denominator to ensure numerical stability. The parameters γ and β are learned during training. Their function is to make sure that the layer is not restricted in what it can represent. For example, if the original activations (without batch normalisation) were optimal, the model is capable of converting the batch normalisation step into the identity mapping by learning $\gamma = \sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}$ and $\beta = \mu_{\mathcal{B}}$.

Batch normalisation was mainly developed to improve optimisation, by making sure that the layers of the network do not have to deal with the so-called *covariate shift*, which is the phenomenon of changing distributions of the activations as the parameters adapt during training. This covatiate shift slows down the training of neural network models due to the burden of having to adapt the parameters of their layers to the changing output distributions of the activations of the previous layer.

Different examples are randomly chosen for inclusion in the minibatches at each pass through the training data and the statistics (mean and variance) differ between the minibatches. This causes some noise to be introduced due to the same examples causing a slightly varying behaviour of the network during each pass through the training data. This phenomenon produces a regularising effect.

Dropout is a regularisation strategy that is mainly applied to fully connected layers. It constitutes the random removal of hidden units and their connections during the training phase, with a specified probability p. The idea behind this seemingly detrimental action is that it prevents the units from co-adapting too much (Srivastava et al., 2014). An interesting interpretation of dropout is that it trains an ensemble consisting of all sub-networks that can be formed by removing units from specified layers of an underlying base network (Goodfellow et al., 2016).



(a) Standard Neural Net (b) After applying dropout.

Figure 4.8: The dopout regularisation method (Srivastava et al., 2014).

Early stopping, which essentially stops the training process when a maximum performance on a validation set is achieved, is also considered to be a regularisation method. The effective model capacity is controlled by limiting the amount of parameter update steps it can take to fit the training data.

4.3.3 Neural network architectures

Fully connected layers

Fully connected layers are the oldest neural network architecture type. A classical neural network consisting of only fully connected layers is called a multilayer perceptron (MLP). A MLP consists of an input layer, an output layer and one or more hidden layers. All units of the input layer are linearly combined with weights and subsequently undergo a non-linear transformation by an activation function for each hidden unit. The resulting outputs of the first hidden layer are the inputs of the next one and so on. A MLP with two hidden layers is illustrated in Figure 4.9. Consecutive hidden layers can be thought of as handling increasing levels of abstraction with regard to the prediction problem. One can think of a neural network as learning a more suitable representation of the data in each consecutive layer. hidden layers can be thought of as single vector-to-vector functions, or as many units that act in parallel, each representing a vector-to-scalar function.

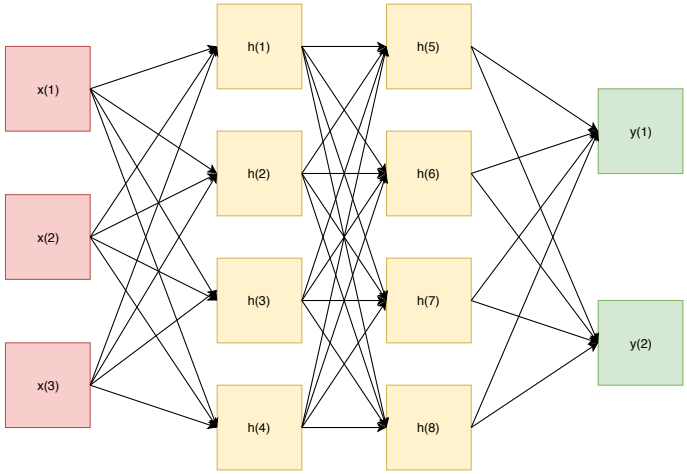


Figure 4.9: Multilayer perceptron with two hidden layers. Three input unites are each connected to the four hidden units of the first hidden layer. These hidden units are connected to the 4 hidden units of the second hidden layer, which are connected to the two output units.

Convolutional layers

Convolutional neural networks (CNN) (Le Cun et al., 1990) are specialized in analysing data with a grid-like structure, with their main application being state-of-the-art image recognition.

Convolutional layers are designed to preserve information on the spatial structure of images (or other grid-like data sources). A convolution in the context of deep learning can be described as a kernel (also called a filter) of learned weights sliding over the data with a certain stride length while computing dot products with data chunks of equal dimensions. The result is an activation map of the input data. Typically, multiple kernels are used to create several activation maps (all contributing a unit of depth or channel to the output). Each of these kernels can be thought of as looking for a specific template or concept in the input data. If the input data is comprised of multiple channels (as is the case with RGB-images in computer vision), the dot products performed by the kernel must extend the full depth of the amount of input channels.

A CNN is a sequence of convolutional layers stacked on top of each other and interspersed with activation functions, followed by one or more fully connected layers. The activation functions allow for the flexibility of non-linear transformations in the mapping of the data between two consecutive layers, which contributes to the versatility of representations that can be learned as the data is processed through the network. In addition to the aforementioned components of a CNN, pooling layers are frequently encountered in CNNs, although their requirement for the CNNs performance is subject to criticism (Springenberg et al., 2014). Pooling layers reduce the size of data representations and equivalently the number of parameters to be learned (the computational burden of the model) by summarising groups of adjacent intermediary outputs into a single value, typically by taking the average or the maximum.

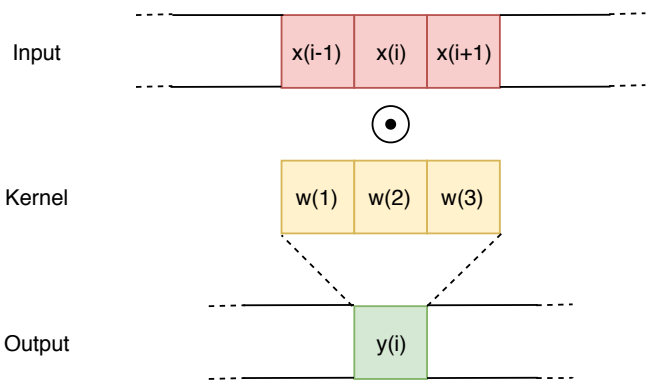


Figure 4.10: The convolution operation in its simplest form with one 3x1 kernel. the value of $y_{(i)}$ is given by $x_{(i-1)}w_{(1)} + x_{(i)}w_{(2)} + x_{(i+1)}w_{(3)}$.

Recurrent layers

Recurrent neural networks (RNNs) (Giles et al., 1994) are an extension of feed-forward neural networks to include feedback connections. This is useful in sequence modeling, as information regarding the previous inputs in a sequence can be stored in the model. When an input vector is fed to a RNN, it is used together with a previous hidden state vector to calculate a new hidden state vector and an output vector. The parameters of functions used to calculate hidden states and outputs are all learned during training. Figure 4.11 is a general illustration of a RNN.

The sharing of parameters resulting from the common hidden state calculation procedure throughout the analysed sequence grants RNNs the possibility of generalising to sequence lengths not seen during training. The training process of a recurrent neural network is similar to training any other neural network, with in this case the gradients flowing through the computational graph of the unrolled representation of the network during backpropagation (termed 'backpropagation through time'). Due to the issue of vanishing or exploding gradients (Hochreiter, 1998) in the classical RNNs, solutions have been developed (LSTMs, GRUs and others) (Hochreiter and Schmidhuber, 1997) (Chung et al., 2014) to mitigate this issue. In multi-layered RNNs, the hidden state vector of the first RNN layer is the input to the next RNN layer and the output is calculated from the hidden state of the last layer.

An additional extension of RNNs for the analysis of sequential data from both sides are bidirectional RNNs (Schuster and Paliwal, 1997), in which the hidden state is computed by concatenation of two hidden states from the forward and backward direction. Unfortunately this results in a doubling of the amount of parameters to be estimated.

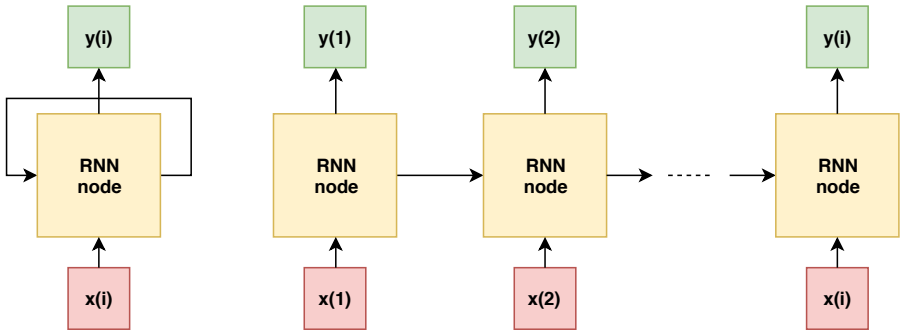


Figure 4.11: Diagram of a single-layer recurrent neural network (left) with unrolled representation (right).

CHAPTER 5

M6A DETECTION USING BASECALLER OUTPUT

5.1 Introduction

The goal here is to develop a model that is capable of accurate detection of N6-methyladenosine without using the raw nanopore signal. The idea is that the output of a known ONT basecaller contains the necessary information for the annotation of modified ribonucleotides. Sequence information and information about the quality of the basecalls are among the features that would be used to train the classifier. The use of sequence information is effective if m6A is found in specific sequence contexts (other than the well known RRACH motif) that have not yet been elucidated. The use of basecaller quality information (i.e. phred scores) could contribute to the model's performance, as the basecaller is expected to be more unsure of its predicted basecalls in the neighbourhood of modified bases. The basecalling algorithm that

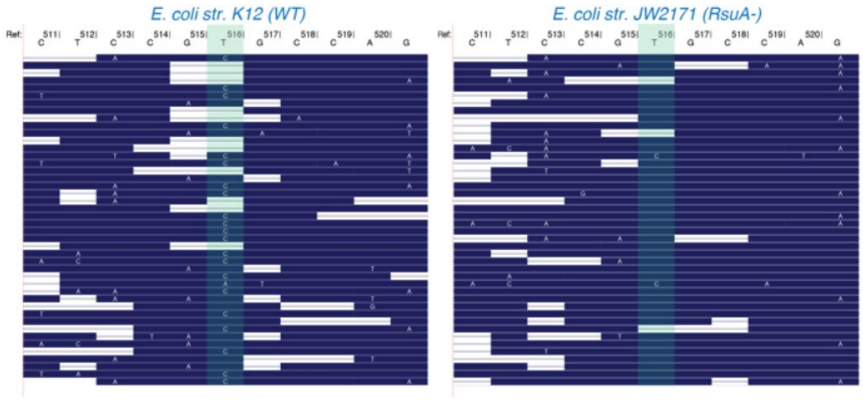


Figure 5.1: A visual example of Basecaller output being examined for the detection of modified RNA bases (Smith et al., 2017). Detection of 7mG modifications in E. coli 16S rRNA. The Genome browser on the left displays an E. coli strain that has a 7mG modification, while the right genome browser shows the same region in the transcriptome for a mutant strain that is incapable of catalysing that particular RNA modification.

has been used on the nanopore direct RNA sequencing data is Albacore v2.1. This

recently developed basecaller differs from previously developed basecallers (with the exeption of Chiron (Teng et al., 2018)) in that it no longer includes a segmentation step that splits the raw nanopore signal into kmer events. The raw nanopore signal is directly plugged into the RNN model, which uses an end-to-end learning approach that was first described in speech recognition applications (Graves and Jaitly, 2014). The complete signal and the complete nucleotide sequence are considered per training example rather than discrete events that are assumed to each correspond to one or two bases. This end-to-end learning approach has led to a significant increase in basecalling performance, as event segmentation introduces some noise and bias into the data. To my knowledge, only one other attempt has been made to detect m6A from basecaller output features. Liu et al. (2019) very recently used support vector machine models to annotate m6A in synthetically modified RNA molecules with an AUC score of around 88%.

5.2 Nanopore data processing

Five direct RNA sequencing runs of the human transcriptome were performed at the University of British Columbia. These sequencing runs all used the R9.4 chemistry. The FAST5 files as well as the basecalled FASTQ files of the runs were downloaded and used for this project. An overview of the data processing pipline from the basecalled reads to the BAM file that was used to generate the data is displayed in Figure 5.2. The quality control plots generated by fastQC, which justify the trimming of the reads before aligning them, can be found in Appendix A.

5.3 Data generation

The first step towards generating the data is obtaining the positions of m6A sites. The MethylTranscriptome database was used for this (see Section 2.4). Pysam, a wrapper for Samtools in Python, was used to extract the reads that were aligned to the genomic positions that correspond to the m6A sites in MetDB. As the effect of modified bases on the nanopore signal manifests itself in the neighbourhood of the m6A positions (Stoiber et al., 2017), a window of 101 bases that is centered at the m6A positions is considered. For all m6A positions in MetDB, the CIGAR strings of the aligned reads were parsed. To annotate the m6A positions, insertions were removed from the reads, while skips and deletions were replaced by dashes. This was done in order to get corresponding positions of the reads and the reference, as is the case in a genome browser. The same was done for the quality scores. The

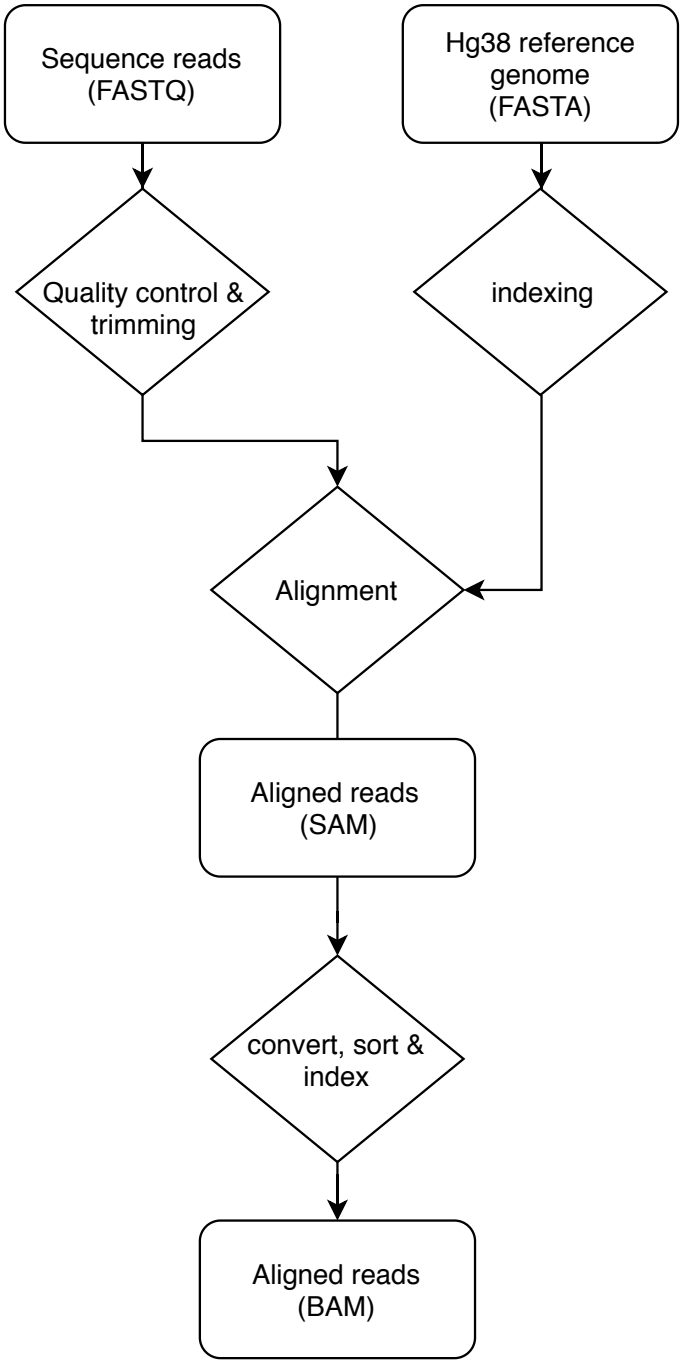


Figure 5.2: Nanopore data processing pipeline. The rounded rectangles represent files and the diamonds represent processes applied to files. First the quality of the FastQ files was assessed using FastQC. From this analysis, it was observed that trimming was required. The first 9 bases were removed using TRIMMOMATIC. The hg38 human reference genome assembly was indexed using minimap2, after which the reads were aligned. Minimap2 was also used for the alignment. Samtools (Li et al., 2009) was used to further process the SAM files. The SAM files were converted to the less memory intensive BAM files, sorted and indexed. The resulting BAM files could be loaded and manipulated in Python to generate the data (see Section 5.3).

DNA reference sequence has been one-hot encoded, which means that each position of the nucleotide sequence was replaced by a vector of length 4, with a one at a single position that depends on the base (A,C,T or G) and zeros at the other three positions. Anti-sense reads' reference sequences were reverse-complemented before inclusion into the data generation process. The amount of reads that did not display a skip or deletion relative to the reference at each of the 101 positions of the window was recorded as well as the mean and standard deviation of the quality scores of the reads. The GTF file associated with the hg38 reference genome was processed to extract information on the chromosomal coordinates of exons and introns, which was also used as a feature for the classification. Finally, the fraction of reads that exactly matches the reference sequence of each position was included. Figure 5.3 is an illustration of what the resulting generated data looks like.

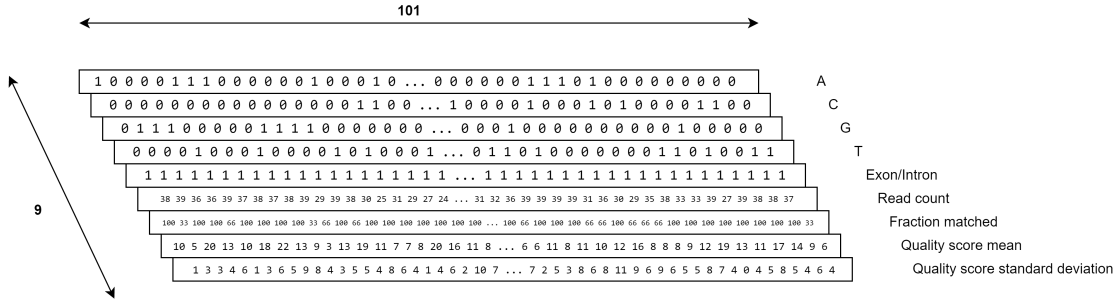


Figure 5.3: One of the hundreds of thousands of generated data instances. The first four channels (rows) are the one-hot encoded sequence. The fifth channel indicates whether the position in the sequence is exonic. The four remaining channels are the amount of reads that aligned to the reference, the fraction of aligned reads that match the reference, the Phred score mean and standard deviation, respectively.

The entries of MetDB correspond to positive examples (i.e. m6A modifications). In order to train a model to detect modified bases, a negative set is also required. It is also important that the negative and positive sets are as similar as possible, except where it matters (in this case differences in features associated with the presence of modified bases). Otherwise the classifier would simply learn unimportant differences that do not have any value when applying the model to independent data. To this end, a negative set was generated by first removing genomic positions that are present in the positive set, then performing a RRACH-motif search. As over 95% of the positive set has its RRACH motif in an exonic region, this constraint was also enforced during the generation of negative examples. After procuring the positions for the negative set, a data generating procedure that is identical to the one used to obtain the positive set was executed. An overview of the complete data generating procedure can be seen in Figure 5.4. Besides the arrays, meta-data was also generated and saved for all instances. This meta-data consists of the path to where each array is saved,

the coverage (read count), genomic coordinates, mean error fraction, mean mapping quality of the reads (not to be confused with the basecalling quality), label, mean read intronic fraction, strand, score (see Section 2.4) and the amount of m6A experiments that report the m6A site. These meta-data were used to load the data and to constrain the loaded data to specified ranges of sufficient quality. The correctness of the generated data was manually confirmed using the Integrative Genomics Viewer (IGV) genome browser (Thorvaldsdottir et al., 2013). An illustration of a generated positive example and its associated screen in IGV can be found in Appendix C.

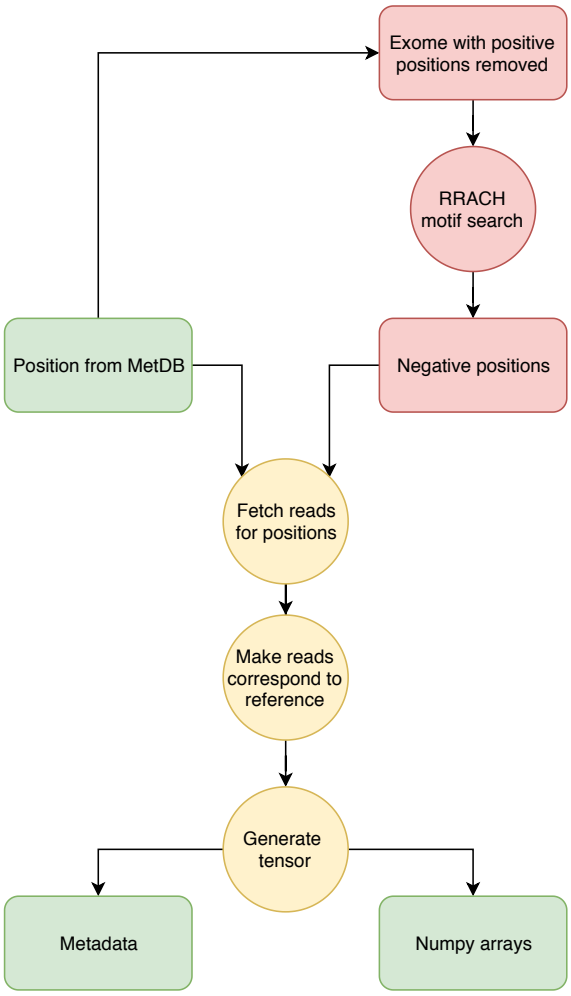


Figure 5.4: A high level overview of the code workflow to generate the positive and negative sets of the data. The positive positions are used to fetch reads from the BAM file that align at the m6A sites and to edit the hg38 reference within a window of 101 around these positions to 'N's before conducting a motif search to obtain negative positions. From the negative positions, only those with their RRACH motif in an exonic region were retained. Once all the reads for the positive and negative positions were fetched, their CIGAR strings were parsed and made to correspond to the reference sequence by removing parts that are present in the reads but not in the reference. Finally, the arrays were created, as well as metadata that will be used for loading and filtering the data.

5.4 Model architecture

The large amount of high-dimensional data with a grid-like topology suggests that a relatively simple convolutional neural network should do the trick. The CNN was built in PyTorch (Paszke et al., 2017). The model architecture is displayed in Figure 5.5. It has three convolutional layers and two fully connected layers. In each of the convolution layers, $10x1$ kernels were used and batch normalisation (see Section 4.3.2) was applied. Twelve of these kernels were used to convolve over the 9 input channels. 8 kernels were used to convolve over the resulting activation maps in the second convolutional layer. The third layer used 6 kernels. The output of the third convolutional layer was fed to two fully connected layers with drouput regularisation before the softmax output (in this case of binary classification this is simply a sigmoid unit). The inclusion of both dropout and batch normalization leads to a significant increase in performance. Increasing the amount of convolution layers does not lead to an improvement of the model's performance. Due to a slight class imbalance, a weighted cross-entropy loss function was used together with an adam optimizer. The model was trained for 40 epochs. Separate training, validation and test datasets were used. After each epoch, the performance on the validation set was evaluated and the model that performed best on this validation set was selected and evaluated on the test set. Training was done on chromosomes 1, 8, 15, 18, 20, 21, 22 and X, validation was done on chromosome 9 and testing was done on chromosome 10. Additional testing was done on chromosome 14, and a withheld split of the data from the chromosomes of the training set. All of these testing procedures yielded similar results.

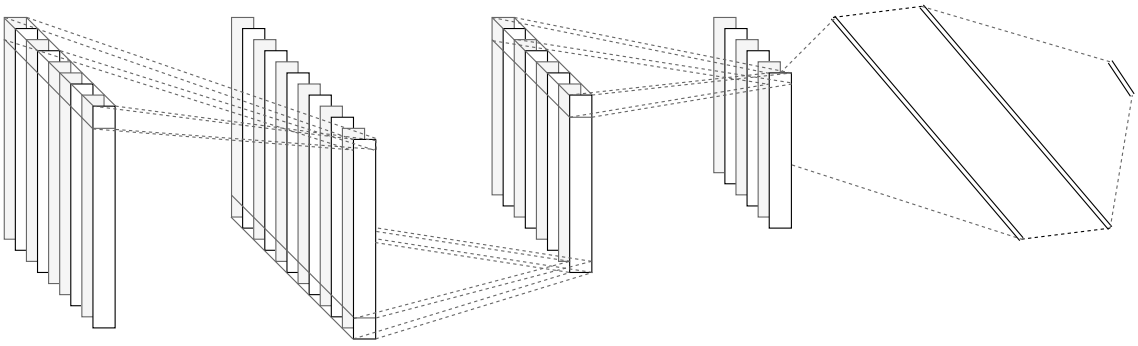


Figure 5.5: Convolutional neural network used for the binary classification of the data. A more detailed depiction of the 9 input channels is shown in Figure 5.3.

5.5 Discussion and model assessment

The first thing that should be investigated is differences in the positive and negative sets. Some data exploration of the positive and negative sets was done to assess the degree of similarity between them. As stated earlier, these differences should be minimal, unless they can be explained by the presence or absence of m6A.

The equivalence of the reads with the reference to which they were mapped was evaluated for the positive and negative sets in Figure 5.6.

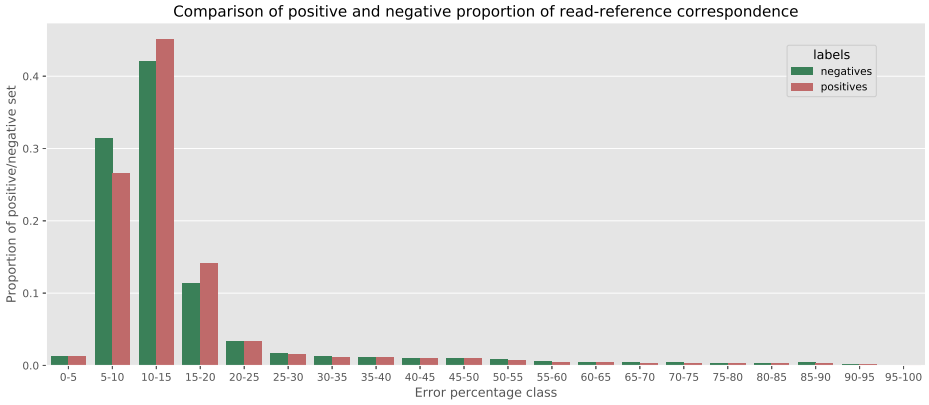


Figure 5.6: Error rates of around 10% are expected for Oxford Nanopore sequencing. The fact that they are slightly higher here may be due to the fact that 1D RNA sequencing is less accurate than the more generally used 2D DNA sequencing. The positive and negative sets have quite similar error distributions, although the reads of the positive set tend to have somewhat higher error rates. This is actually expected due to the less certain behaviour of the basecaller when it encounters a modified nucleotide.

An interesting phenomenon can be observed when comparing the base occurrence in the 101 windows with ACTG-plots. It can be seen in Figure 5.7 that m6A sites correspond to regions of the genome with a relatively higher percentage of G and C, when compared to the other exonic RRACH motifs in the genome. The total percentages of A, C, T and G in the genome are 30%, 20%, 30% and 20% respectively. This is about the same as what can be seen in the ACTG-plot of the negative set.

Figure 5.8 compares the fraction of the aligned reads that match the reference per position, the average quality scores and their standard deviation. A magnification of this figure between positions 45 and 55 (near the RRACH motif) can be found in Appendix F.

Some assumptions were made during the development of this model. The first assumption that was made is concerned with the origin of the m6A positions in the pos-

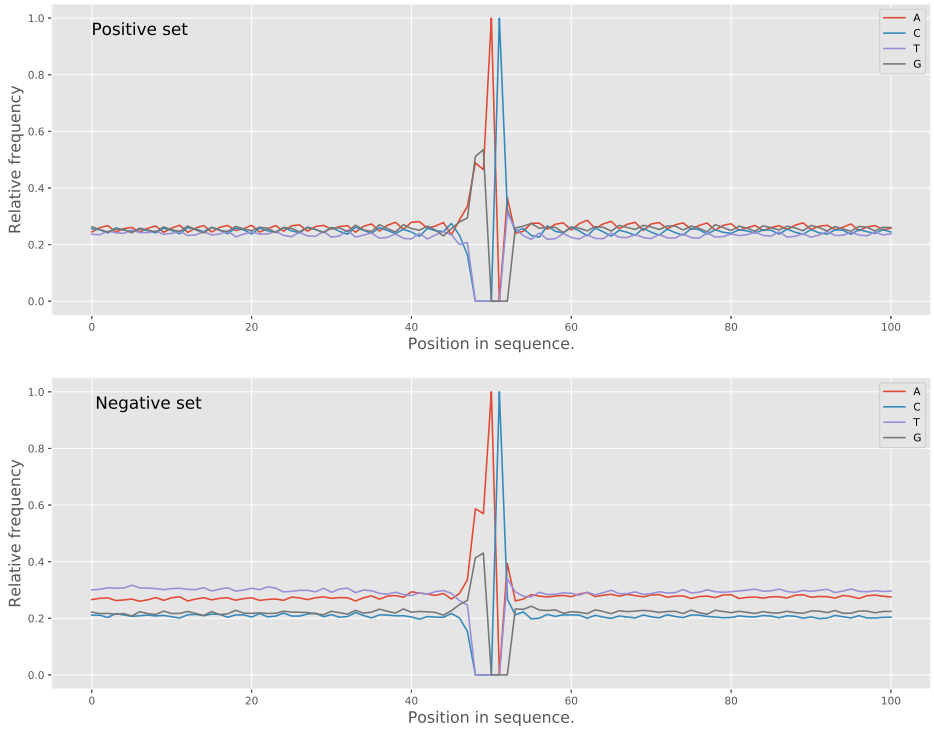


Figure 5.7: **ACTG plots**, summarizing the frequency of each nucleotide at each position for the positive (top) and negative (bottom) sets. The RRACH motifs can be observed at positions 48-52. There appears to be a difference between the sequence distributions of m6A sites and those of other motifs in exonic regions. This is reminiscent of the different frequency of occurence of the DNA methylation 5mC in CpG islands (Caiafa and Zampieri, 2005). A and T are less prevalent near m6A sites. The frequency of occurence of the different RRACH motifs also seems to be different at m6A sites, where G is favored over A in the first two positions.

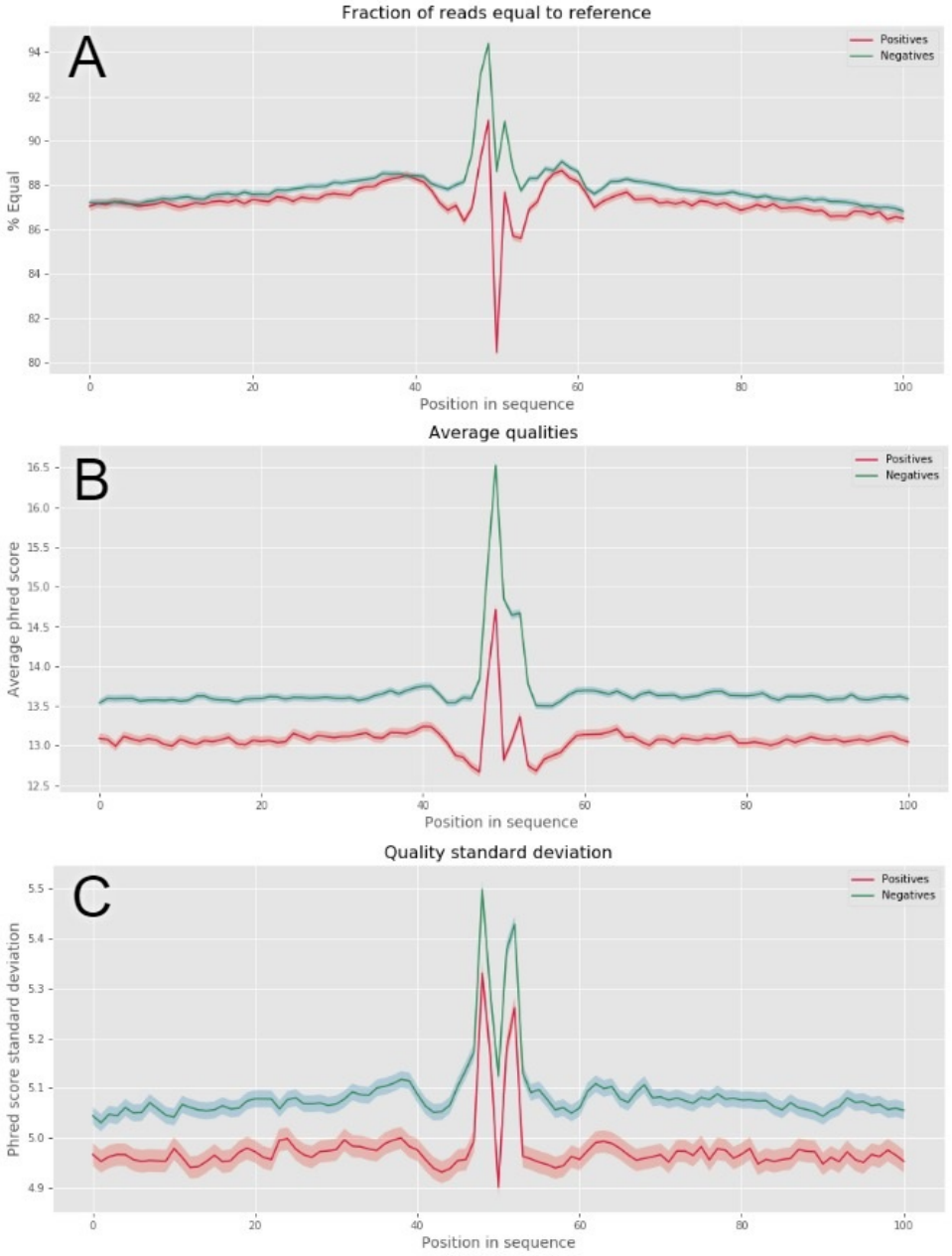


Figure 5.8: **(A)** Fraction of reads that do not have deletions or mismatches per position. In general this is lower in the positive set, this is especially the case near the m6A sites. This was expected due to the less certain behaviour of the basecaller when it encounters modified bases. There is a very pronounced drop at the m6A position. The reason that this same drop can be seen in the negative set (albeit much less pronounced) may be due to the presence of false negatives in the negative set. **(B)** Average mean qualities per position. There is a baseline average phred score difference of about 0.5 at large distances from the m6A site. This baseline difference is due to the difference in nucleotide distribution of the positive and negative sets, as Albacore calls the four bases with different error probabilities (Krishnakumar et al., 2018). The difference is considerably larger at the m6A sites. This corroborates the hypothesis that the basecaller is less certain of its basecalls in the neighbourhood of the m6A sites. **(C)** Average standard deviations of the quality scores. These are lower in the positive set. 95% confidence intervals are included in the plots. In the three displayed feature plots, peaks can be seen in the neighbourhood of the RRACH motifs. This is due to the fact that the reads were specifically selected to have the RRACH motif so fewer mismatches are expected, without taking the presence of m6A into account.

itive set. As mentioned earlier these were downloaded from the MetDB single base database (see Section 2.4). The validity of the procedure used in MetDB to obtain single nucleotide resolution m6A positions by linking the chromatin immunoprecipitation peaks to the nearest RRACH motif seems reasonable, although a limited amount of false positives are to be expected. A second assumption is that these m6A positions which originate from independent human m6A-seq experiments, are sufficiently conserved in the human RNA samples that were subjected to the Oxford Nanopore direct RNA-sequencing runs. M6A modifications have been reported to be quite conserved within species (Schwartz et al., 2013), although some subsets of m6A sites exhibit tissue specificity and are subject to dynamic regulation. Some false positives may arise due to this assumption as well. A third important assumption concerns the absence of m6A sites in the negative set. The positions of the negative set were obtained by searching for RRACH motifs in the exome with positive m6A positions removed. Although the RRACH motif is about two orders of magnitude more prevalent than m6A modifications (Dominissini et al., 2012), some false negatives are expected in the negative set.

These assumptions have been avoided in other m6A detection research by using synthetically modified RNA molecules, although the capacity of models trained on these RNA molecules to generalise to real modified RNA molecules has not yet been demonstrated. Other ways to reduce the reliance on these assumptions would be to conduct m6A-sequencing and direct RNA sequencing experiments on the same sample or to make use of conventional single-nucleotide resolution methods (see Section 2.3). In this work, a stringent filtering of the data was conducted. See Appendix B for the filtering criteria and their distributions after filtering was applied. These distributions are similar in the positive and negative sets.

When all features of the generated and filtered data were used to train the model, an ROC-AUC score of 0.87 and an average precision score of 0.83 were achieved on the test set. The associated ROC-curve and PR-curve are displayed in Figure 5.9. The model was additionally tested on chromosome 14 and an independent test set from the same chromosomes as the training set, where AUC scores of 0.86 and 88 were achieved, respectively. Overfitting of the model to chromosome-specific noise does not appear to be an issue.

In order to interpret feature importance, the model was retrained and re-evaluated using different independent subsets of the features. Additionally, the model was trained and evaluated on all features but using a reduced window size of 31 instead of 101. The results of these experiments can be seen in Figure 5.10.

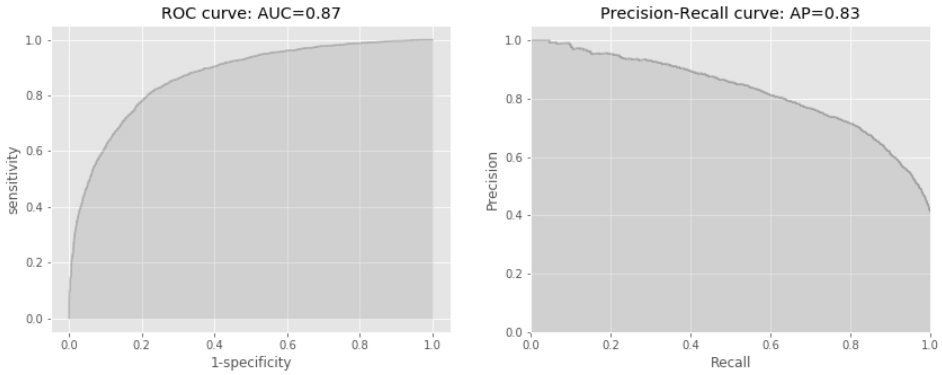


Figure 5.9: **ROC curve** (left) and **PR-curve** (right) of the model displayed in Figure 5.5 with inclusion of all features, trained on the filtered data from chromosomes 1, 8, 15, 18, 20, 21, 22 and X, validated on chromosome 9 and tested on chromosome 10.

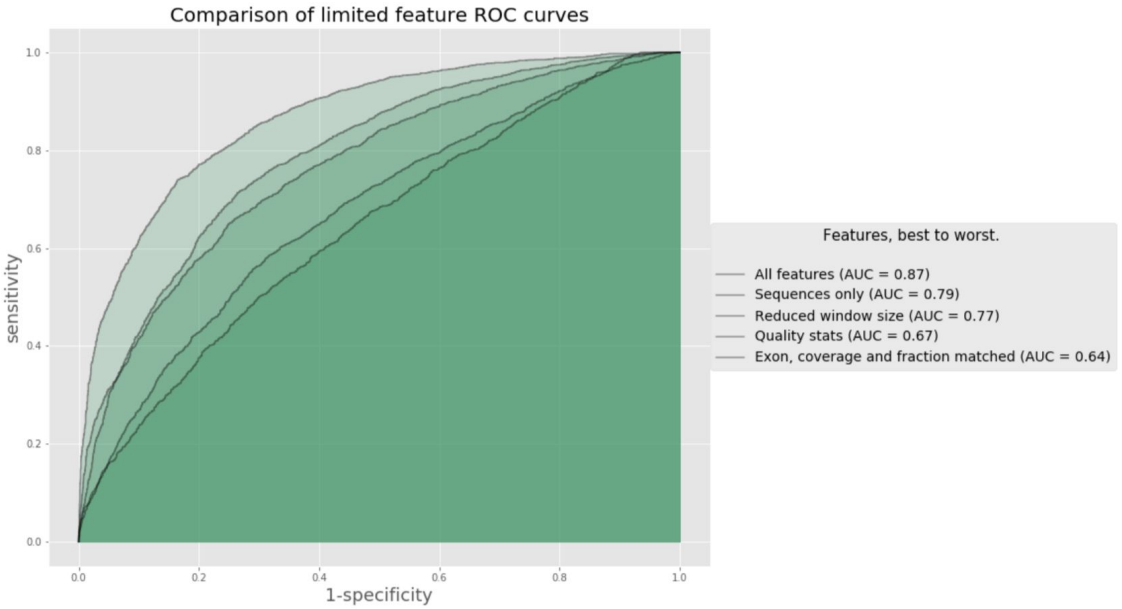


Figure 5.10: **ROC curves** of the CNN model using either all features, only sequence features, a reduced window size, quality features or a combination of exon presence, coverage and the fraction of reads equal to the reference sequence per position.

The sequence information is most useful in predicting the presence of m6A sites, followed by the quality information and the three other features.

In order to further investigate the validity of the model assumptions, the arrays that were erroneously predicted were examined using ACTG plots similar to Figure 5.7. These plots can be inspected in Appendix F. The sequence distributions suggest that the negative set arrays may be contaminated by positives and vice versa.

As the amount of independent studies confirming the presence of m6A sites at specific positions in the genome increases, the amount of false predictions of a resulting model due to wrong labels decreases. An experiment to validate this claim is to split the positive set into several equal parts, each with an increasing amount of independent studies reporting the m6A positions, before training the model on each of the parts. The performance should increase with the amount of independent studies reporting the m6A sites. This experiment was done using the model described in Section 5.4. The positive set was first filtered in the same way as described in Appendix B, without filtering on the amount of independent samples reporting the m6A sites. Afterwards, it was sorted by the amount of samples and split into four equal parts (each having around fourteen thousand instances). The first part contained the m6A sites reported by up to three independent experiments, the m6A sites in the second part were reported by between three and ten independent experiments. The third part contained m6A sites reported by between 10 and 35 experiments and the final part contained between 35 and 52 experiments. Due to the equal sizes of the parts, differences in performance cannot be attributed to differences in the amount of data that is provided to train the model. The resulting constant increase in performance as more independent studies confirm m6A sites can be seen in Figure 5.11.

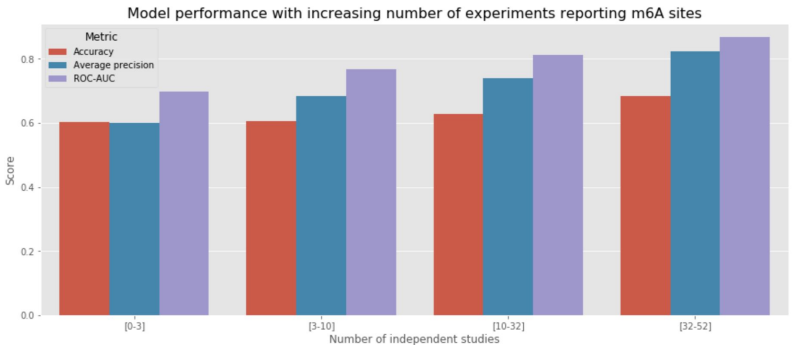


Figure 5.11: Barplot displaying the increase in performance of the CNN model described in Section 5.4 as the amount of independent studies reporting the m6A positions in the positive set increases. It is clear that the three chosen performance metrics (accuracy, average precision and AUC) improve from left to right.

CHAPTER 6

FUTURE PERSPECTIVES AND CONCLUSIONS

6.1 Future perspectives

6.1.1 Improvement and expansion of the basecaller output model

The most important improvement to be made is on the data. The main shortcomings of the data that was currently used and some suggestions for improvement were discussed in the previous chapter. A more general improvement would be the use of the newest R9.5.1 chemistry in the Oxford Nanopore sequencing experiments. ONT is constantly improving the quality of their nanopore proteins and reagents, leading to more accurate nucleic acid translocation measurements, resulting in an increased performance of the predictive models for basecalling.

The synthetic m6A data from liu et al (Liu et al., 2019), can be used either for independent testing or improvement of the model. The ideal situation would be to conduct experiments where a sample is split in two parts, one of which undergoes a single nucleotide resolution method for m6A detection, while the other is subjected to a direct RNA sequencing run. That way one can be much more certain of the sample-specific m6A sites and their precise positions.

While the focus of this thesis was on m6A, a similar method can be applied for the detection of other RNA modifications, when sufficient high-resolution data of their positions is available.

As the only sample-specific requirement of the model is a fastQ file, it could also be used for m6A detection of reads originating from another long-read sequencing technology such as Pacific Biosciences' SMRT sequencing platform.

Potential improvements of the model itself are the inclusion of features originating from the signal itself, such as the mean and standard deviation of the signal segments that correspond to each base prediction of the basecaller.

6.1.2 Development of a m6A-augmented basecaller

A m6A detection model that is based on the output of another predictive model (e.g. Albacore) has its drawbacks. These drawbacks include the rather large amount of computation steps between the input direct RNA sequencing data and the m6A site predictions, as the reads first need to be basecalled by Albacore, aligned and processed further before being fed to the model. Furthermore, the basecaller output model makes a single prediction per RRACH motif, neglecting differentially-modified transcripts.

It would be advantageous to develop and train a new basecaller with addition of the m6A class to the four other base classes. This would allow for real-time m6A analysis without the burden of much intermediary computation and without the potential inclusion of the biases of another basecaller. There are two main approaches for the development of such a basecaller: one that works on signal data that has been segmented into events and one that makes predictions directly on the raw signal in an end-to-end fashion.

Post-event segmentation basecaller

Until recently, the go-to method to basecall Oxford Nanopore signals was to first segment the signal into events using a Poremodel provided by ONT. Basecallers using this strategy include Metrichor, Nanonet, DeepNano (Boža et al., 2017) , NanoCall (David et al., 2017) and early versions of Albacore. After event segmentation and a limited data processing, a HMM or RNN model is used to classify the events into bases.

Raw signal basecaller

As event segmentation is known to introduce some noise and bias, a significant increase in performance is achieved by avoiding this step entirely and building a classifier that works directly with the raw signals. Recently developed basecallers, such as Chiron (Teng et al., 2018), Wavenano (Wang et al., 2018) and the latest versions of Albacore are examples of the raw signal approach. Chiron employs a convolutional recurrent neural network (Shi et al., 2015), where residually-connected convolutions (He et al., 2015) learn to extract a useful feature representation from the signal, which

is then passed to bidirectional LSTM layers. In order to address the differing output sequence lengths associated with signal segments of specified length, a connectionist temporal classification decoder (Graves et al., 2006) is used as loss function and output decoder. An overview of the required data processing to annotate Oxford Nanopore signals with bases (including m6A) is described in Appendix E. Figure 6.1 is an illustration of a raw signal that has been annotated with the aforementioned data processing procedure.

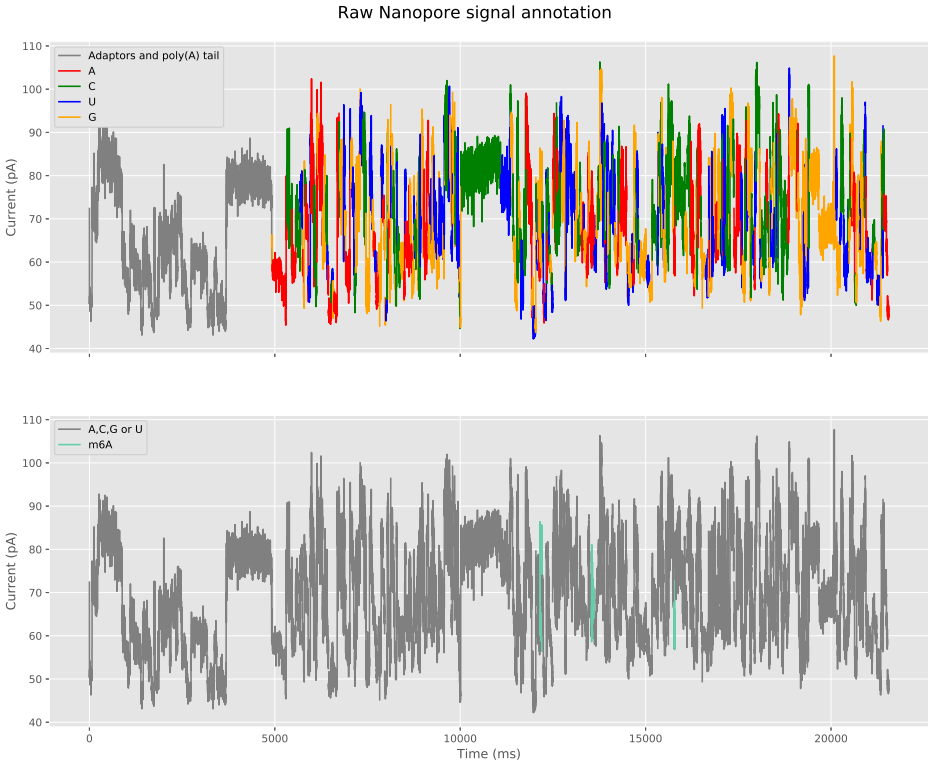


Figure 6.1: MinION signal measurement that has been annotated with its corresponding conventional bases (top), and with m6A (bottom). Adaptors and polyA tails are largely removed by ONT's segmentation algorithm. Sometimes the first events are contaminated by adaptor sequence measurements. This explains the overall lower quality of the first basecalls, which can be seen in Appendix A.

6.2 Conclusions

M6A sites seem to reside in specific sequence contexts that are enriched in C and G. To my knowledge this has not yet been reported or tested in other studies. Nor can it be learned by analysing synthetically modified mRNA constructs. The convolutional neural network model developed in this thesis has the potential to generalize to other long-read sequencing technologies and to become effective in the epitranscriptome analysis of mammalian biological samples. Furthermore, it only requires a FastQ file as input from the sequenced sample. Ideally, data of higher quality should be procured and used to further improve the model and potentially achieve state-of-the-art m6A detection performance. With a data processing pipeline for the annotation of raw Nanopore signals in place, a foundation has been laid for the development of novel m6A detection models that address the drawbacks of a basecaller output model.

BIBLIOGRAPHY

Alarcón, C. R., Goodarzi, H., Lee, H., Liu, X., Tavazoie, S., and Tavazoie, S. F. (2015). HNRNPA2B1 is a mediator of m6A-dependent nuclear RNA processing events. *Cell*, 162(6):1299–1308.

Ardui, S., Ameur, A., Vermeesch, J. R., and Hestand, M. S. (2018). Single molecule real-time (SMRT) sequencing comes of age: Applications and utilities for medical diagnostics. *Nucleic Acids Research*, 46(5):2159–2168.

Batista, P. J. (2017). The RNA Modification N6-methyladenosine and Its Implications in Human Disease. *Genomics, Proteomics & Bioinformatics*, 15(3):154–163.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York.

Boccaletto, P., Machnicka, M. A., Purta, E., Piątkowski, P., Bagiński, B., Wirecki, T. K., de Crécy-Lagard, V., Ross, R., Limbach, P. A., Kotter, A., Helm, M., and Bujnicki, J. M. (2018). MODOMICS: A database of RNA modification pathways. 2017 update. *Nucleic Acids Research*, 46(D1):D303–D307.

Bolger, A. M., Lohse, M., and Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics*, 30(15):2114–2120.

Boža, V., Brejová, B., and Vinař, T. (2017). DeepNano: Deep recurrent neural networks for base calling in MinION nanopore reads. *PLOS ONE*, 12(6):e0178751.

Caiafa, P. and Zampieri, M. (2005). DNA methylation and chromatin structure: The puzzling CpG islands. *Journal of Cellular Biochemistry*, 94(2):257–265.

Chiu, C.-C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., Kannan, A., Weiss, R. J., Rao, K., Gonina, E., Jaitly, N., Li, B., Chorowski, J., and Bacchiani, M. (2017). State-of-the-art Speech Recognition With Sequence-to-Sequence Models. *arXiv:1712.01769 [cs, eess, stat]*.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:1412.3555 [cs]*.

Cock, P. J. A., Fields, C. J., Goto, N., Heuer, M. L., and Rice, P. M. (2010). The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Research*, 38(6):1767–1771.

David, M., Dursi, L. J., Yao, D., Boutros, P. C., and Simpson, J. T. (2017). Nanocall: An open source basecaller for Oxford Nanopore sequencing data. *Bioinformatics (Oxford, England)*, 33(1):49–55.

Deng, X., Chen, K., Luo, G.-Z., Weng, X., Ji, Q., Zhou, T., and He, C. (2015). Widespread occurrence of N6-methyladenosine in bacterial mRNA. *Nucleic Acids Research*, 43(13):6557.

Dominissini, D., Moshitch-Moshkovitz, S., Salmon-Divon, M., Amariglio, N., and Rechavi, G. (2013). Transcriptome-wide mapping of N6-methyladenosine by m6A-seq based on immunocapturing and massively parallel sequencing. *Nature Protocols*, 8(1):176–189.

Dominissini, D., Moshitch-Moshkovitz, S., Schwartz, S., Salmon-Divon, M., Ungar, L., Osenberg, S., Cesarkas, K., Jacob-Hirsch, J., Amariglio, N., Kupiec, M., Sorek, R., and Rechavi, G. (2012). Topology of the human and mouse m6A RNA methylomes revealed by m6A-seq. *Nature*, 485(7397):201–206.

El Yacoubi, B., Bailly, M., and de Crécy-Lagard, V. (2012). Biosynthesis and Function of Posttranscriptional Modifications of Transfer RNAs. *Annual Review of Genetics*, 46(1):69–95.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34.

Garalde, D. R., Snell, E. A., Jachimowicz, D., Sipos, B., Lloyd, J. H., Bruce, M., Pantic, N., Admassu, T., James, P., Warland, A., Jordan, M., Ciccone, J., Serra, S., Keenan, J., Martin, S., McNeill, L., Wallace, E. J., Jayasinghe, L., Wright, C., Blasco, J., Young, S., Brocklebank, D., Juul, S., Clarke, J., Heron, A. J., and Turner, D. J. (2018). Highly parallel direct RNA sequencing on an array of nanopores. *Nature Methods*, 15(3):201–206.

Giles, C., Kuhn, G., and Williams, R. (1994). Dynamic Recurrent Neural Networks - Theory and Applications. *Ieee Transactions on Neural Networks*, 5(2):153–156. WOS:A1994NH76800001.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of Machine Learning Research*, 9:8.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep Sparse Rectifier Neural Networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Graves, A., Fernández, S., and Gomez, F. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *In Proceedings of the International Conference on Machine Learning, ICML 2006*, pages 369–376.

Graves, A. and Jaitly, N. (2014). Towards End-to-End Speech Recognitionwith Recurrent Neural Networks. *Proceedings of Machine Learning Research*, page 9.

Grozhik, A. V., Linder, B., Olarerin-George, A. O., and Jaffrey, S. R. (2017). Mapping m6A at Individual-Nucleotide Resolution Using Crosslinking and Immunoprecipitation (miCLIP). *Methods in Molecular Biology (Clifton, N.J.)*, 1562:55–78.

Harcourt, E. M., Ehrenschwender, T., Batista, P. J., Chang, H. Y., and Kool, E. T. (2013). Identification of a Selective Polymerase Enables Detection of N ^{6} -Methyladenosine in RNA. *Journal of the American Chemical Society*, 135(51):19079–19082.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*.

Hochreiter, S. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67.

Horiuchi, K., Umetani, M., Minami, T., Okayama, H., Takada, S., Yamamoto, M., Aburatani, H., Reid, P. C., Housman, D. E., Hamakubo, T., and Kodama, T. (2006). Wilms' tumor 1-associating protein regulates G2/M transition through stabilization of cyclin A2 mRNA. *Proceedings of the National Academy of Sciences*, 103(46):17278–17283.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2016). Densely Connected Convolutional Networks. *arXiv:1608.06993 [cs]*.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167 [cs]*.

Jia, G., Fu, Y., Zhao, X., Dai, Q., Zheng, G., Yang, Y., Yi, C., Lindahl, T., Pan, T., Yang, Y.-G., and He, C. (2011). *N*6-Methyladenosine in nuclear RNA is a major substrate of the obesity-associated FTO. *Nature Chemical Biology*, 7(12):885–887.

Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

Krishnakumar, R., Sinha, A., Bird, S. W., Jayamohan, H., Edwards, H. S., Schoeniger, J. S., Patel, K. D., Branda, S. S., and Bartsch, M. S. (2018). Systematic and stochastic influences on the performance of the MinION nanopore sequencer across a range of nucleotide bias. *Scientific Reports*, 8.

Le Cun, Y., Jackel, L. D., Boser, B., Denker, J. S., Graf, H. P., Guyon, I., Henderson, D., Howard, R. E., and Hubbard, W. (1990). Handwritten Digit Recognition: Applications of Neural Net Chips and Automatic Learning. In Soulié, F. F. and Hérault, J., editors, *Neurocomputing*, pages 303–318. Springer Berlin Heidelberg, Berlin, Heidelberg.

Levy, S. E. and Myers, R. M. (2016). Advancements in Next-Generation Sequencing. *Annual Review of Genomics and Human Genetics*, 17(1):95–115.

Li, H. (2018). Minimap2: Pairwise alignment for nucleotide sequences. *Bioinformatics (Oxford, England)*, 34(18):3094–3100.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., and 1000 Genome Project Data Processing Subgroup (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics (Oxford, England)*, 25(16):2078–2079.

Liu, H., Begik, O., Lucas, M. C., Mason, C. E., Schwartz, S., Mattick, J. S., Smith, M. A., and Novoa, E. M. (2019). Accurate detection of m6A RNA modifications in native RNA sequences. *bioRxiv*.

Liu, H., Wang, H., Wei, Z., Zhang, S., Hua, G., Zhang, S.-W., Zhang, L., Gao, S.-J., Meng, J., Chen, X., and Huang, Y. (2018). MeT-DB V2.0: Elucidating context-specific functions of N6-methyl-adenosine methyltranscriptome. *Nucleic Acids Research*, 46(D1):D281–D287.

Liu, J., Yue, Y., Han, D., Wang, X., Fu, Y., Zhang, L., Jia, G., Yu, M., Lu, Z., Deng, X., Dai, Q., Chen, W., and He, C. (2014). A METTL3-METTL14 complex mediates mammalian nuclear RNA N6-adenosine methylation. *Nature Chemical Biology*, 10(2):93–95.

Liu, N. and Pan, T. (2015). RNA epigenetics. *Translational Research*, 165(1):28–35.

Liu, N., Parisien, M., Dai, Q., Zheng, G., He, C., and Pan, T. (2013). Probing N6-methyladenosine RNA modification status at single nucleotide resolution in mRNA and long noncoding RNA. *RNA (New York, N.Y.)*, 19(12):1848–1856.

Lusser, A., editor (2017). *RNA Methylation: Methods and Protocols*. Number 1562 in Methods in Molecular Biology. Humana Press, New York. OCLC: ocn961002872.

Marimont, R. B. and Shapiro, M. B. (1979). Nearest Neighbour Searches and the Curse of Dimensionality. *IMA Journal of Applied Mathematics*, 24(1):59–70.

McElwee, S. (2018). *Probabilistic Clustering Ensemble Evaluation for Intrusion Detection*. PhD thesis.

Meng, J., Lu, Z., Liu, H., Zhang, L., Zhang, S., Chen, Y., Rao, M. K., and Huang, Y. (2014). A protocol for RNA methylation differential analysis with MeRIP-Seq data and exomePeak R/Bioconductor package. *Methods (San Diego, Calif.)*, 69(3):274–281.

Meyer, K. D., Saletore, Y., Zumbo, P., Elemento, O., Mason, C. E., and Jaffrey, S. R. (2012). Comprehensive analysis of mRNA methylation reveals enrichment in 3' UTRs and near stop codons. *Cell*, 149(7):1635–1646.

Mullis, K., Faloona, F., Scharf, S., Saiki, R., Horn, G., and Erlich, H. (1986). Specific enzymatic amplification of DNA in vitro: The polymerase chain reaction. *Cold Spring Harbor Symposia on Quantitative Biology*, 51 Pt 1:263–273.

Paszke, A., Gross, S., and Lerer, A. (2017). Automatic differentiation in PyTorch.

Piekna-Przybylska, D., Decatur, W. A., and Fournier, M. J. (2008). The 3D rRNA modification maps database: With interactive tools for ribosome analysis. *Nucleic Acids Research*, 36(Database issue):D178–D183.

Ping, X.-L., Sun, B.-F., Wang, L., Xiao, W., Yang, X., Wang, W.-J., Adhikari, S., Shi, Y., Lv, Y., Chen, Y.-S., Zhao, X., Li, A., Yang, Y., Dahal, U., Lou, X.-M., Liu, X., Huang, J., Yuan, W.-P., Zhu, X.-F., Cheng, T., Zhao, Y.-L., Wang, X., Danielsen, J. M. R., Liu, F., and Yang, Y.-G. (2014). Mammalian WTAP is a regulatory subunit of the RNA N6-methyladenosine methyltransferase. *Cell Research*, 24(2):177–189.

Quick, J., Loman, N. J., Duraffour, S., Simpson, J. T., Severi, E., Cowley, L., Bore, J. A., Koundouno, R., Dudas, G., Mikhail, A., Ouédraogo, N., Afrough, B., Bah, A., Baum, J. H. J., Becker-Ziaja, B., Boettcher, J. P., Cabeza-Cabrerizo, M., Camino-Sánchez, Á., Carter, L. L., Doerrbecker, J., Enkirch, T., Dorival, I. G., Hetzelt, N., Hinzmann, J., Holm, T., Kafetzopoulou, L. E., Koropogui, M., Kosgey, A., Kuisma, E., Logue, C. H., Mazzarelli, A., Meisel, S., Mertens, M., Michel, J., Ngabo, D., Nitzsche, K., Pallasch, E., Patrono, L. V., Portmann, J., Repits, J. G., Rickett, N. Y., Sachse, A., Singethan, K., Vitoriano, I., Yemanaberhan, R. L., Zekeng, E. G., Racine, T., Bello, A., Sall, A. A., Faye, O., Faye, O., Magassouba, N., Williams, C. V., Amburgey, V., Winona, L., Davis, E., Gerlach, J., Washington, F., Monteil, V., Jourdain, M., Bererd, M., Camara, A., Somlare, H., Camara, A., Gerard, M., Bado, G., Baillet, B., Delaune, D., Nebie, K. Y., Diarra, A., Savane, Y., Pallawo, R. B., Gutierrez, G. J., Milhano, N., Roger, I., Williams, C. J., Yattara, F., Lewandowski, K., Taylor, J., Rachwal, P., J. Turner, D., Pollakis, G.,

Hiscox, J. A., Matthews, D. A., Shea, M. K. O., Johnston, A. M., Wilson, D., Hutley, E., Smit, E., Di Caro, A., Wölfel, R., Stoecker, K., Fleischmann, E., Gabriel, M., Weller, S. A., Koivogui, L., Diallo, B., Keïta, S., Rambaut, A., Formenty, P., Günther, S., and Carroll, M. W. (2016). Real-time, portable genome sequencing for Ebola surveillance. *Nature*, 530(7589):228–232.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Sanger, F., Nicklen, S., and Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, 74(12):5463–5467.

Schneider, T. D. and Stephens, R. M. (1990). Sequence logos: A new way to display consensus sequences. *Nucleic Acids Research*, 18(20):6097–6100.

Schuster, M. and Paliwal, K. (Nov./1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Schwartz, S., Agarwala, S. D., Mumbach, M. R., Jovanovic, M., Mertins, P., Shishkin, A., Tabach, Y., Mikkelsen, T. S., Satija, R., Ruvkun, G., Carr, S. A., Lander, E. S., Fink, G. R., and Regev, A. (2013). High-Resolution Mapping Reveals a Conserved, Widespread, Dynamic mRNA Methylation Program in Yeast Meiosis. *Cell*, 155(6):1409–1421.

Shi, B., Bai, X., and Yao, C. (2015). An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. *arXiv:1507.05717 [cs]*.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

Smith, A. M., Jain, M., Mulroney, L., Garalde, D. R., and Akeson, M. (2017). Reading canonical and modified nucleotides in 16S ribosomal RNA using nanopore direct RNA sequencing. *bioRxiv*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for Simplicity: The All Convolutional Net. *arXiv:1412.6806 [cs]*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:30.

Stoiber, M. H., Quick, J., Egan, R., Lee, J. E., Celniker, S. E., Neely, R., Loman, N., Pennacchio, L., and Brown, J. B. (2017). De novo Identification of DNA Modifications Enabled by Genome-Guided Nanopore Signal Processing. *bioRxiv*.

Teng, H., Cao, M. D., Hall, M. B., Duarte, T., Wang, S., and Coin, L. J. M. (2018). Chiron: Translating nanopore raw signal directly into nucleotide sequence using deep learning. *GigaScience*, 7(5).

Thorvaldsdottir, H., Robinson, J. T., and Mesirov, J. P. (2013). Integrative Genomics Viewer (IGV): High-performance genomics data visualization and exploration. *Briefings in Bioinformatics*, 14(2):178–192. WOS:000316694700006.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need.

Vilfan, I. D., Tsai, Y.-C., Clark, T. A., Wegener, J., Dai, Q., Yi, C., Pan, T., Turner, S. W., and Korlach, J. (2013). Analysis of RNA base modification and structural rearrangement by single-molecule real-time detection of reverse transcription. *Journal of Nanobiotechnology*, 11(1):8.

Wang, S., Li, Z., Yu, Y., and Gao, X. (2018). WaveNano: A signal-level nanopore basecaller via simultaneous prediction of nucleotide labels and move labels through bidirectional WaveNets. *Quantitative Biology*, 6(4):359–368.

Wang, X., Lu, Z., Gomez, A., Hon, G. C., Yue, Y., Han, D., Fu, Y., Parisien, M., Dai, Q., Jia, G., Ren, B., Pan, T., and He, C. (2014). N6-methyladenosine-dependent regulation of messenger RNA stability. *Nature*, 505(7481):117–120.

Wang, X., Zhao, B. S., Roundtree, I. A., Lu, Z., Han, D., Ma, H., Weng, X., Chen, K., Shi, H., and He, C. (2015). N(6)-methyladenosine Modulates Messenger RNA Translation Efficiency. *Cell*, 161(6):1388–1399.

Weirather, J. L., de Cesare, M., Wang, Y., Piazza, P., Sebastiano, V., Wang, X.-J., Buck, D., and Au, K. F. (2017). Comprehensive comparison of Pacific Biosciences and Oxford Nanopore Technologies and their applications to transcriptome analysis. *F1000Research*, 6.

Wolpert, D. and Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

Zheng, G., Dahl, J. A., Niu, Y., Fedorcsak, P., Huang, C.-M., Li, C. J., Vågbø, C. B., Shi, Y., Wang, W.-L., Song, S.-H., Lu, Z., Bosmans, R. P. G., Dai, Q., Hao, Y.-J., Yang, X., Zhao,

W.-M., Tong, W.-M., Wang, X.-J., Bogdan, F., Furu, K., Fu, Y., Jia, G., Zhao, X., Liu, J., Krokan, H. E., Klungland, A., Yang, Y.-G., and He, C. (2013). ALKBH5 is a mammalian RNA demethylase that impacts RNA metabolism and mouse fertility. *Molecular Cell*, 49(1):18–29.

Zhu, P. and Craighead, H. G. (2012). Zero-mode waveguides for single-molecule analysis. *Annual Review of Biophysics*, 41:269–293.

APPENDIX A

DIRECT RNA SEQUENCING RUNS AND QUALITY CONTROL

Table A.1: Additional information on the five direct RNA sequencing runs.

Sequencing run	Count	Length range	Average length	Average %GC
UBC run 1	587569	12-13138	1050	63
UBC run 2	534410	19-12606	985	63
UBC run 3	751775	15-18505	1053	63
UBC run 4	259368	6-13355	910	63
UBC run 5	628824	11-18293	1228	64

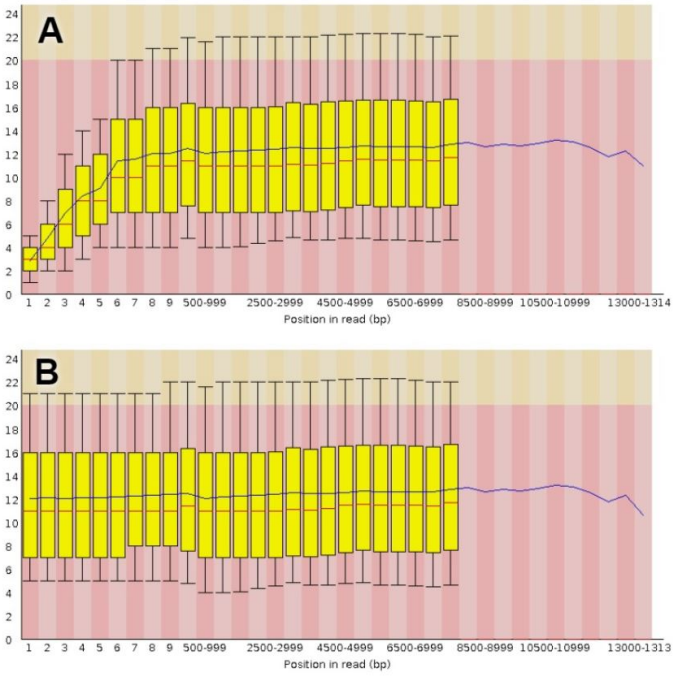


Figure A.1: Comparison of the quality scores across all bases before **(A)** and after **(B)** trimming off the first 9 bases is displayed for UBC run 1. The other sequencing runs give rise to similar plots. The Figures were generated by fastQC. A description of how these quality scores are calculated can be found in section 3.3.

APPENDIX B

DISTRIBUTIONS OF SELECTED DATA AFTER FILTERING

B.1 Positive set

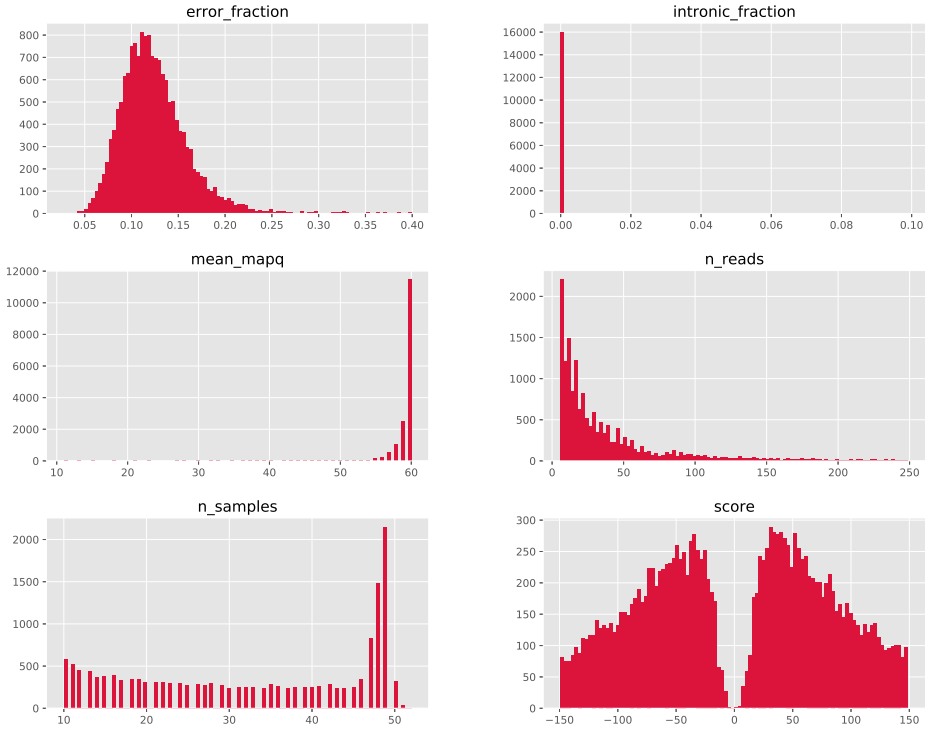


Figure B.1: **Positive set** filtered data distributions. The imposed constraints are as follows: the average fraction of error (i.e. mismatches) of the reads must be below 0.4, the overlap of the considered windows with intronic regions is limited to 10% of the window size, the mean mapping quality of the reads must exceed 10, at least 5 reads must be aligned at the motif, the m6a site positions must be corroborated by at least two independent m6A-seq experiments (see section 2.3) and the distance of the ChIP peak from the nearest RRACH motif may not exceed 150 (see section 2.4).

B.2 Negative set

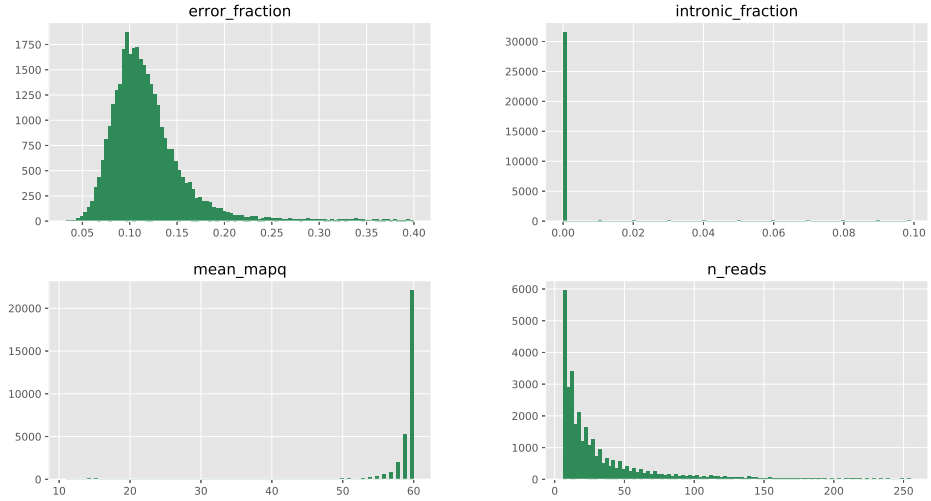


Figure B.2: **Negative set** filtered data distributions. The same constraints were applied as with the positive set, with the exceptions of the amount of experiments *n_samples* and the *score*, which are specific to the positive set.

APPENDIX C

MANUAL EVALUATION OF DATA GENERATION

A m6A position in the MetDB (the database with m6a sites, see section 2.4) with an amount of aligned reads that is feasible to manually examine, was randomly sampled. This sampled position was 29326365 on chromosome 21. The validity of the generated data is confirmed by comparing a part of the generated array with the corresponding part in the IGV genome browser. The RRACH motif is the region that will be subjected to manual evaluation. It is indicated by rectangles in the figures C.1 and C.2 below.

Figure C.1: Integrative Genomics Viewer with the RRACH region of interest indicated by a rectangle. A total amount of 5 reads are aligned to this region, of which the sequence is AGACT. This region lies within the *BACH1* gene and is exonic. When hovering over the aligned reads, the quality scores per position can be seen. In in the selected region these are $\{13, 13, 4, 21, 9\}$, $\{10, 9, 4, 20, 9\}$, $\{13, 13, 5, 20, 13\}$, $\{13, 11, 5, 14\}$ and $\{8, 9, 5, 4, 9\}$ respectively. When we calculate the means $\mu = \frac{1}{N}\sum_1^N x_i$ of these qualities per position and round the result to the nearest integer, we get $12, 10, 13, 11$ and 7. Next, the standard deviations $\sigma = \sqrt{\frac{1}{N-1}\sum_1^N (x_i - \mu)^2}$ are calculated. The resulting values (rounded to the nearest integer) are $6, 6, 5, 4$ and 2. The read counts per position are 5, except at the second last position, where it is 4 due to a deletion in the last aligned read. As the first, fourth and fifth positions have either a mismatch or a deletion, the fractions matched per position is 80%, 100%, 100%, 80%, and 80%.

With these calculations, upon inspection of figure C.2, it can be concluded that the information in the genome browser corroborates the values in the generated array and the data generation was successful.

	A	C	G	T	Exon/Intron	Read Count	Fraction matched	Quality mean	Quality stdev
40	1	0	0	0	1	5	100	26	7
41	1	0	0	0	1	5	100	26	4
42	1	0	0	0	1	5	100	16	4
43	0	0	0	1	1	5	100	14	3
44	0	0	1	0	1	5	100	17	6
45	0	0	0	1	1	5	80	14	1
46	0	0	0	1	1	5	100	15	4
47	0	1	0	0	1	5	80	10	6
48	1	0	0	0	1	5	80	12	6
49	0	0	1	0	1	5	100	10	6
50	1	0	0	0	1	5	100	13	5
51	0	1	0	0	1	4	80	11	4
52	0	0	0	1	1	5	80	7	2
53	0	1	0	0	1	5	100	13	2
54	0	1	0	0	1	5	100	11	4
55	0	0	0	1	1	5	100	6	3
56	0	1	0	0	1	5	80	8	2
57	1	0	0	0	1	5	40	11	5
58	0	0	1	0	1	5	100	17	3
59	0	0	0	1	1	5	100	17	2
60	0	0	1	0	1	5	100	19	2

Figure C.2: The middle part of the generated array is displayed, with the central RRACH morif region indicated by a rectangle. The one-hot sequence encoding is equivalent to AGACT.

APPENDIX D

BIO-INFORMATICS COMMANDS

D.1 Albacore basecalling

```
read_fast5_basecaller.py -i .../Fast5/ -t N_threads -s .../output_dir/
 -f FLO-MIN106 -k SQK-RNA001 -c r941_70bps_rna_linear.cfg -o fast5,fastq
```

D.2 Trimmomatic

```
java -jar .../trimmomatic-0.38.jar SE -phred33 .../reads.fastq
reads_trim.fastq HEADCROP:9
```

D.3 FastQC

```
fastqc reads.fastq
```

D.4 Minimap2

```
.../minimap2 -k14 -d hsa_genome.mmi .../genome.fa
../minimap2 -ax splice -uf -k14 -t N_threads -secondary=no .../hsa_genome.mmi
reads_trimmed.fastq > reads_trimmed.sam
```

D.5 Samtools

```
samtools view -bS -o reads_trimmed.bam reads_trimed.sam
samtools sort -@ 10 -m 1000M -o reads_sorted_trimmed.bam reads_trimmed.bam
samtools index reads_sorted_trimmed.bam
```

APPENDIX E

ANNOTATING THE RAW NANOPORE SIGNAL

In order to train a novel Oxford Nanopore basecaller that is capable of predicting the presence of m6A in addition to the four unmodified regular bases, a suitable model should be provided with parts of signals, as well as the base labels that correspond to these parts.

What follows is a brief description of the workflow diagram depicted in Figure E.1. The Fast5 files, originating from a MinION device, are basecalled. The output is a FastQ file with the basecalled reads and new Fast5 files with event segmentation information and the raw signals. Table E.1 provides an overview of the event segmentation information that is available in the Fast5 files. M6A positions in MetDB are filtered according to a threshold of their score values (see Section 2.4) and the amount of independent studies reporting them. The FastQ file undergoes a similar procedure as described in Section 5.2. The CIGAR string of the resulting Bam file is parsed and used to annotate the reads with m6A positions. Corrections to Albacore's basecalls can also be applied during this step. The 'move' and 'start' event characteristics are used to assign base labels to specific segments of the raw signal.

Table E.1: Event segmentation information contained in basecalled fast5 files.

Event characteristic	**Description**	**example**
Mean	Scaled mean of the signal segment	114.14
Start	Starting position of the signal segment	10347
Standard deviation	Standard deviation of the signal segment	19.82
Length	Length of the signal segment	15
Model state	State of ONT's pore model	CCAAA
Move	Amount of bases translocated	2

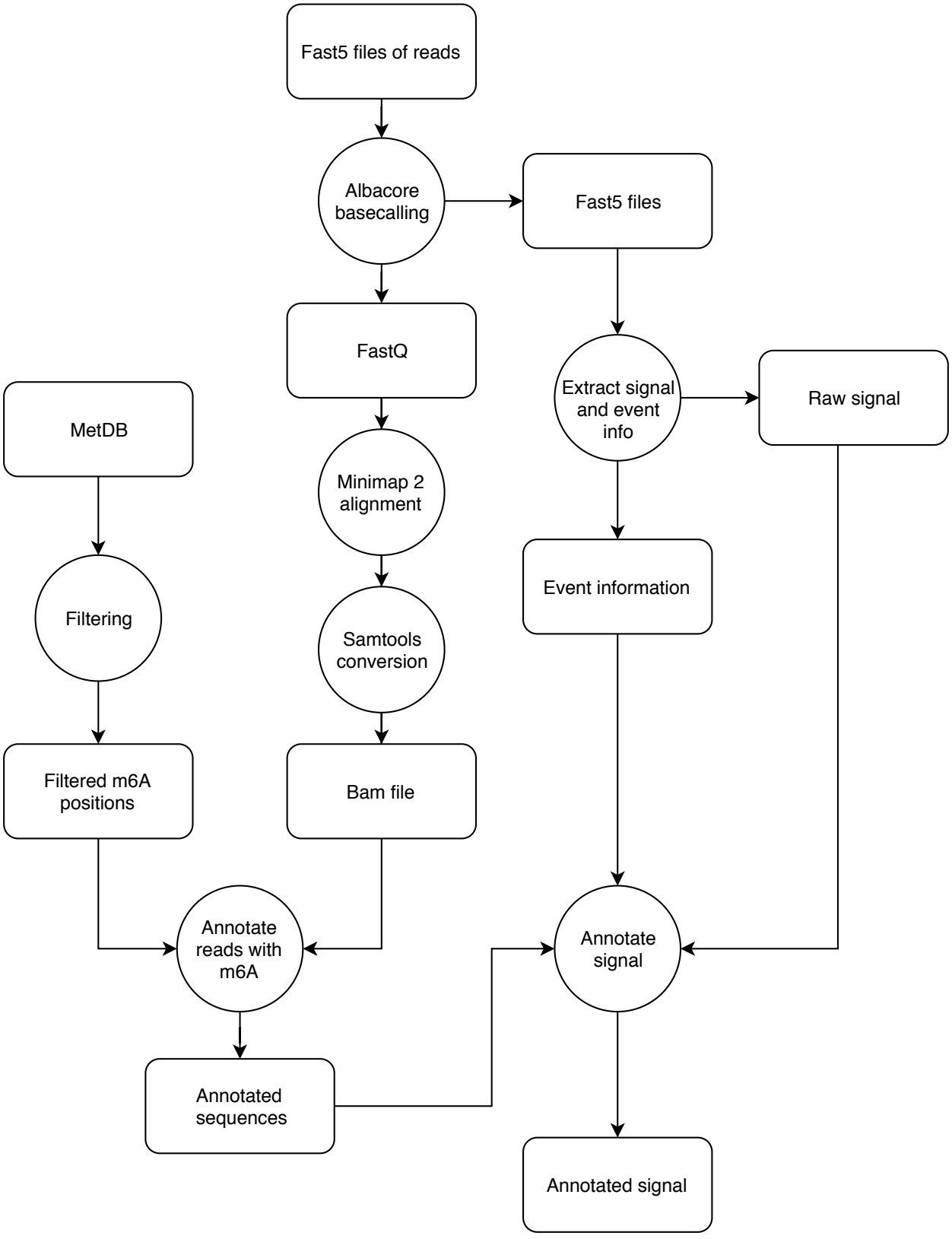


Figure E.1: Base annotation of Oxford Nanopore signals. Rounded rectangles denote files or a description of their content and circles designate processes applied to these files.

APPENDIX F

SUPPLEMENTARY FIGURES

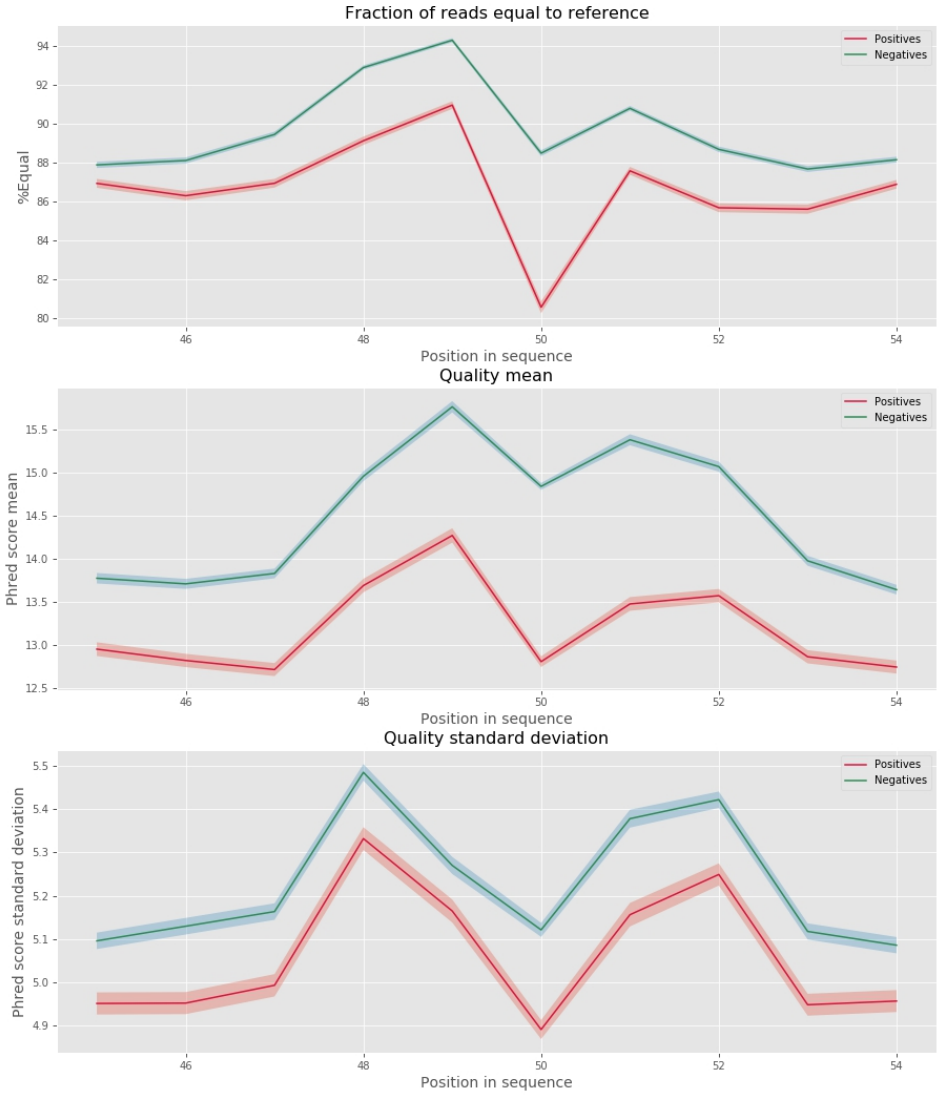


Figure F.1: Magnification of Figure 5.8 near the RRACH motif.

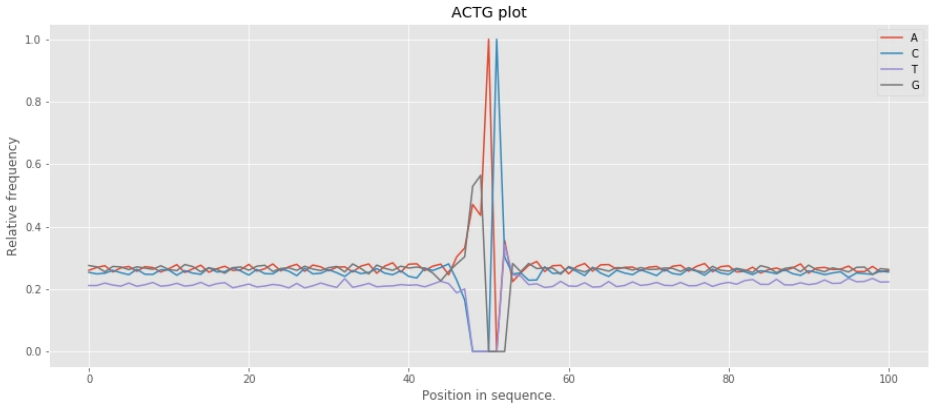


Figure F.2: **Top 5% wrong positive predictions ACTG plot**. The CNN model described in Section 5.4 predicted (with a probability of at least 0.95) that the arrays summarised in this ACTG plot originate from the positive set, although they have negative labels. The similarity of this plot to the ACTG plot of the positive set in Figure 5.7 is inline with the proposition that the negative set is contaminated by some positives.

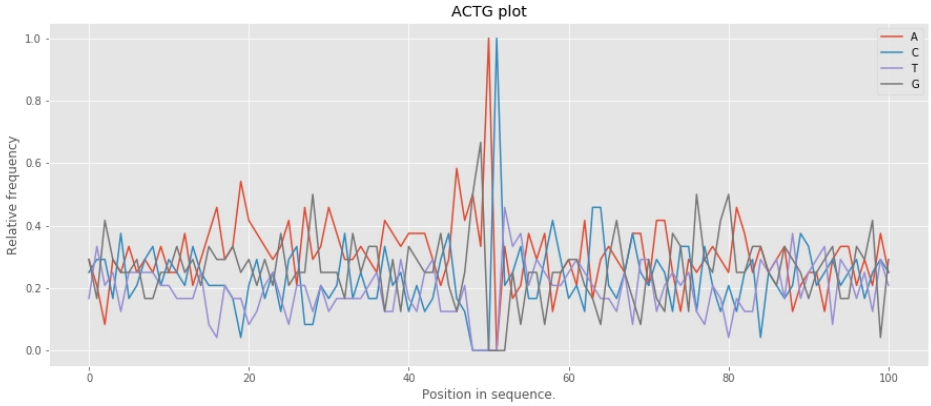


Figure F.3: **Top 5% wrong negative predictions ACTG plot**. The CNN model described in Section 5.4 predicted (with a probability of at least 0.95) that the arrays summarised in this ACTG plot originate from the negative set, although they have positive labels.