Using Machine Learning to Predict and Better Understand DNA Methylation and Genomic Enhancers

Matthew R. Huska

Dissertation zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Berlin, 2018

Erstgutachter: Prof. Dr. Martin Vingron
Zweitgutachter: em. Prof. Dr. Joachim Selbig

Tag der Disputation: 15.12.2017

Preface

Acknowledgements

There are many people who have made my time as a PhD student both enjoyable and enlightening, and without whose help the work would not have been possible. First, I would like to thank my supervisor Martin Vingron, for interesting discussions, support and guidance. Additionally, I would like to thank Annalisa Marsico for several fruitful collaborations on both scientific projects and teaching, and also for her friendship. I was also very happy to be a part of the International Max Planck Research School for Computational Biology and Scientific Computing, it was an honor as well as the source of several close friendships. Thanks to Kirsten Kelleher and Fabian Feutlinske for their help and organization over the years, and all the members of the IMPRS past and present who made my PhD so much more than just an educational experience.

I would like to thank Anna Ramisch for working together with me on the enhancer co-training project, it was an intense time and the project wouldn't have been possible without her. I am grateful to Wolfgang Kopp, for being a wonderful office mate, for the many conversations we have shared over the years, and for inviting me to help with the DREAM TF binding challenge. I would like to thank Alessandro Mammana for countless hilarious conversations, interesting scientific discussions, for letting me convince him that he shouldn't spend his whole PhD programming in Java, and for generally being a great friend. I would like to thank Jonathan Göke, Julia Lasserre and Roland Krause for their guidance in the early stages of my PhD. Also, I would like to thank Mahsa Ghanbari, Mike Love, Juliane Perner, Johannes Helmuth, Philipp Benner, Robert Schöpflin, Verena Heinrich, Alena van Bömmel, Edgar Steiger, Xinyi Yang, Matthias Heinig, Ho-Ryun Chung, Sebastiaan Meijsing and all the current and former members of the Vingron Department who were absolutely wonderful colleagues during my time at the MPI. Finally, thanks to Anna Ramisch, Edgar Steiger for their critical reading of parts of the thesis, and to Kirsten Thobe for a critical reading of the entire thesis as well as helping with the German translation of the Summary.

Lastly, a special thanks to my parents, Mike and Sharon, for their support and

understanding.

Publications

The work discussed in this thesis is centered around two major projects that I completed during my PhD. The idea for the project presented in Chapter 4 started with a lively discussion about CpG islands between Martin Vingron and Jose Muino at one of our weekly group meetings, and the realization that I could apply the same methods I was evaluating for my early research into predicting enhancers using genome sequence to the exploration of this new dataset. This work was published in PLoS Computational Biology (Huska and Vingron, 2016). The second project started following a seminar from Martin Vingron about his long term hopes for the study of gene regulation using computational biology, after which myself and several other members of the lab decided to create a working group to create more integrated models of gene regulation rather than separately modeling different parts of the regulatory process. This resulted in the forming of the "Work group for integrated prediction of gene expression", and the project presented in Chapter 5 was the first subproject of that group and was carried out along with Anna Ramisch with supervision and brainstorming from both Martin Vingron and Annalisa Marsico. The project was presented at the German Bioinformatics Conference 2016, and at the same time made available as a preprint article at PeerJ Preprints (Huska et al., 2016).

Matthew R. Huska
Berlin, 2018

Contents

Chapter 1

Introduction

In this thesis we address the general question of how the genome's sequence influences various elements involved in gene regulation. To this end, we employ methods from the field of machine learning and pattern recognition, to identify patterns in DNA that are associated with genomic methylation and enhancers.

We begin with an overview of the biological background (Chapter 2) that motivates the problems that are addressed later in the thesis, and also explains the experimental techniques whose data we rely on. Chapter 3 describes machine learning in general as well as the major topics within the field of machine learning that are relevant to this thesis, with a focus on *machine learning on DNA sequences*. The chapter contains an overview of several machine learning methods that are commonly applied to biological problems involving biological sequences, including those used in this thesis.

Chapter 4 presents the problem of predicting non-methylated regions of the genome using DNA sequence. This study came about because of the development of new experimental methods for measuring DNA methylation genome-wide. These methods shifted the focus from CpG islands (CpG-rich regions of DNA sequence in the genome which were used as a proxy for non-methylated regions) to experimentally determined non-methylated islands (NMIs). Using the data from these new experiments, we use machine learning methods to directly learn the sequence features that are indicative of non-methylated regions in several species and tissues. Specifically, we use a supervised learning called a support vector machine, along with several kernels that are specifically designed for sequential data like DNA sequences.

In Chapter 5 we present the problem of predicting genomic regulatory regions called enhancers. A fundamental step in addressing this problem is identifying a reliable set of known enhancers that we can train our models on. We demonstrate that several experimental methods that are all assumed to be predictive of enhancer regions do not agree with each other, and therefore we use a small set of enhancers that are identified by three separate experimental methods as our starting point. This small set motivates us to use semi-supervised learning techniques, and we combine ChIP-seq data and DNA sequence features as input to the predictive model.

Chapter 2

Biological Background

From the largest whale to the tiniest bacteria, the countless different species on the planet can at first glance seem like they couldn't be more different from each other. Despite these superficial differences, the surprising fact is how similar all of these organisms are at the cellular level. The basic mechanisms and building blocks required for life are shared among all species, and the study and better understanding of these fundamental mechanisms in any organism increases our understanding of all living things.

Many of the fundamental mechanisms required for life depend on just a few important types of molecules: DNA, which makes up our genome, proteins, which form most of the cell's mass and carry out a vast array of cellular functions, and RNA, a temporary copy of portions of the genome that can act as an intermediary between the genome and proteins or can itself be functional (see Figure 2.1). Despite the critical role these molecules play there is still much that we don't know about how they function and interact with each other.

2.1 The genome

The most important molecule for any living organism is its *genome*, a sequence of discrete chemical units forming a structure known as DNA, or deoxyribonucleic acid. This molecule contains the information required for an organism to survive and reproduce, and is physical means of passing this information from an organism to its offspring. The building blocks that make up the genome are called nucleotides, four chemicals which comprise the alphabet that is used to spell out the genome: adenine (A), cytosine (C), guanine (G) and thymine (T). Individual nucleotides are joined together along a chemical backbone made up of deoxyribose, and sequences of these nucleotides spell out different functional regions of the genome. These functional regions include *genes*, as well as *regulatory regions*.

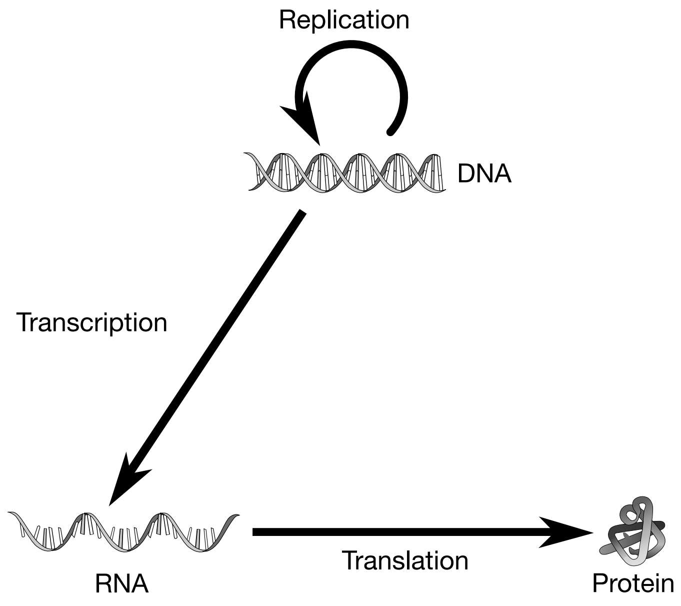


Figure 2.1: The central dogma of molecular biology. DNA is replicated when the cell divides and creates two daughter cells. During transcription, DNA is read and copied into RNA. Translation is the process of synthesizing a protein based on the sequence of an mRNA molecule. From Wikimedia Commons, 2016a.

The genes encode proteins as well as functional RNAs, or ribonucleic acids, which together carry out most of the important functions in the cell. Regulatory regions such as enhancers and insulators modulate the activity of genes.

An organism's genome is contained inside the nucleus of every one of its cells, with a couple of rare exceptions such as mature red blood cells. Despite the fact that these cells may form tissues with drastically different functions and appearances, the genome in all cells of an organism is the same. Another way of saying this is that cells can have different phenotypes (observable characteristics), but the same genotype (genome sequence). It only through tightly controlling the activity of genes and proteins through many mechanisms that different cell types can be created and maintained.

2.1.1 Transcription and translation

Transcription and *translation* are the processes which take the information encoded by a gene and produce a functional molecule from it, either a protein or functional RNA. Transcription involves creating a temporary copy of the gene, and requires a protein complex, called RNA polymerase, to read the nucleotides that make up the gene and create a copy of these nucleotides in a new molecule called RNA. This RNA is a similar molecule to DNA with three exceptions: it has a different chemical backbone (ribose instead of deoxyribose), it is a single stranded molecule instead of a double stranded molecule, and it has a slightly different alphabet consisting of adenine (A), cytosine (C), guanine (G) and uracil (U). RNA can itself have a function (such as microRNAs and lncRNAs), or if it is a messenger RNA (mRNA) it can be further read and translated into a protein. This translation is performed by the ribosome, a complex made up of proteins as well as RNA (ribosomal RNA or rRNA). The ribosome reads the sequence of the mRNA and translates it into a sequence of protein building blocks called amino acids, and this chain of amino acids folds to form a functional protein which can then carry out some role in the cell.

2.1.2 Transcriptional regulation

Transcriptional regulation is one of the most important processes involved in defining the identity and function of cells. Because the genome is identical between all cells, and the basic machinery required for transcription is available in nearly all cells, phenotypic differences are primarily achieved by controlling which genes are transcribed as well as the rate of this transcription. Through this regulation, different cells can specialize to perform different tasks and form distinct tissues such as a muscle, neuron, or kidney. Even small changes in this transcriptional control can increase the risk of diseases such as cancer, diabetes, autoimmune diseases and neurological disorders (Lee and Young, 2013), which makes the study of transcriptional regulation important not just to further our understanding of the basic mechanisms of life, but also for the diagnosis and treatment of disease. Despite this importance, there are still many aspects the mechanisms of transcriptional regulation that are poorly understood, and the analysis presented in this thesis are intended to advance our knowledge of the topic.

2.1.3 Transcription factors

Transcription factors (TFs) are proteins that physically interact with DNA in a sequence-specific manner to regulate transcription. TFs function alone or in complex with other proteins to promote or repress the binding of RNA polymerase, and can bind directly upstream of the transcription start site (promoter proximal), or in other regions (called cis-regulatory regions) which can be distant from the gene being regulated. Since some of these transcription factors are only present and active in a subset of cells, they are able to regulate the transcription of genes in a cell type-specific manner, and therefore have a strong influence on cellular phenotypes.

2.1.4 Enhancers

Enhancers are cis-regulatory regions that are not promoter proximal and that can boost gene expression of distal genes when activated. Enhancers function by looping to the promoter that they regulate, and are activated by sequence specific transcription factors, which create an environment permissive to transcription. It is common that multiple enhancers regulate a gene, and they do not necessarily regulate the closest gene, in some cases acting over several megabases. One such example is an enhancer of the Shh gene, which is over 1 megabase away from the gene's promoter (Amano et al., 2009). Some enhancers also generate short transcripts, called enhancer RNAs or eRNAs (Kim et al., 2010), though the functional role of these transcripts, if any, is not known. Nevertheless, the importance of regulatory elements like enhancers in human disease was suggested years ago by the observation in many genome-wide association studies that causal variants are very often identified distant from transcribed genes (Helgadottir et al., 2007). This suggestion was confirmed in more recent studies showing that the disruption of the function of these regulatory elements can lead to changes in gene expression and disease phenotypes (Weedon et al., 2014; Lupiáñez et al., 2015).

Unlike genes, which generate transcripts and have some well understood DNA sequence features that can be used to identify their location in the genome, most enhancers do not seem to need to generate transcripts to function and have more poorly understood sequence features, and therefore are difficult to identify reliably genome-wide. Also, because of the variable distance between the enhancer and the gene it regulates, even when an enhancer is identified it is difficult to determine its target. These two issues, along with their importance to disease, make enhancers particularly relevant to study.

2.2 The epigenome

Given that the genome of all cells within an organism share the same DNA, it begs the question of what causes some cells to look and behave differently from each other, but causes daughter cells to mostly resemble their parent cell. The answer to this question is that there is an additional layer of information on top of the DNA, known as the *epigenome* (the prefix epi- means over, or outside of) which can influence the activity of genes. The strict definition of epigenetics refers to information outside of the genome that can be passed from one generation to the next, but the more commonly used definition that we will use in this thesis relaxes the heritability constraint. Specifically, we will focus on chemical modifications of the DNA itself, as well as to a subset of proteins called histones.

2.2.1 Histone modifications

In order to fit the genome of eukaryotic organisms into the nucleus of a cell it must be tightly packed. This packing of DNA is performed by protein complexes called nucleosomes, which consist of the DNA wrapped around a set of eight protein subunits called *histones*. The basic structure of a histone consists of a core which interacts with other histones as well as the DNA, as well as a long tail of amino acids. The amino acids that make up this tail can be chemically modified by certain transcription factors, and specific chemical modifications are known to co-locate with different genomic “states” such as active or repressed genes, enhancers, and repressed regions. There are several types of chemical modifications that can be made, including methylation, phosphorylation, acetylation, ubiquitylation, and sumoylation. The most well studied, in the context of gene regulation,

Table 2.1: Commonly studied histone tail modifications.

Histone modification	Genome state
H3K4me1	Enhancer regions
H3K4me2	Permissive euchromatin
H3K4me3	Transcriptional elongation; active euchromatin; promoter regions
H3K27ac	Transcriptional activation
H3K27me3	Transcriptional silencing; X-inactivation; bivalent genes/gene poising
H3K36me3	Transcriptional elongation

are methylation and acetylation, where relationships have been identified between the modification of certain amino acids in the histone's tail and certain genomic regions and transcriptional states (see Table 2.1 for a summary of some of the most commonly measured histone modifications, or Lawrence, Daujat, and Schneider, 2016 for an extensive review).

2.2.2 DNA methylation

It is also possible to directly modify the nucleotides that make up the genome without changing the genome's DNA sequence. One of the most common DNA modifications is *DNA methylation*, which consists of the addition of a methyl group to a cytosine nucleotide. This modification almost exclusively takes place when the cytosine is followed by guanine nucleotide, and the resulting sequence is usually referred to as a CpG dinucleotide to differentiate it from the base paired cytosine and guanine residues that occur between DNA strands.

The effect of DNA methylation is to silence genomic regions, both by repressing the transcription of nearby genes and by throttling the activity of enhancer regions (Magnusson et al., 2015 for one example of the latter). This silencing can be accomplished by a set of proteins that bind methylated DNA, and consequently interact with other proteins that compact the genome.

In almost all vertebrate cells, the majority of CpGs are methylated. Because of this, we tend to focus on the more rare occurrence of non-methylated CpGs. Individual stretches of these regions have been identified and studied for more than

30 years (Cooper, Taggart, and Bird, 1983), but only with newer experimental approaches based on high-throughput sequencing have we finally been able to measure DNA methylation genome-wide.

2.3 Experimental methods for measuring epigenetic modifications

Over the last several years, many experimental methods have been developed to help us identify and study the relationships between epigenetic modifications and transcriptional regulation. These methods all leverage the huge advances that have taken place in the field of high throughput sequencing, and it is only due to these advances that we can now have a picture of epigenetic modifications genome-wide. These genome-wide measurements allow us to develop a more general understanding of these mechanisms and the interplay between genomic sequence, epigenetic modifications, and transcriptional regulation.

2.3.1 Histone modifications: ChIP-seq

In order to measure protein-DNA interactions genome-wide, the most common experimental method by far is *ChIP-seq*, or chromatin immunoprecipitation followed by sequencing. It is useful not only for measuring the binding of specific transcription factors (Robertson et al., 2007; Johnson et al., 2007), but also for measuring histones with specific modifications (Barski et al., 2007).

The method works by first strongly fixing proteins to the DNA, usually using the chemical formaldehyde. Next, the DNA is sheared into smaller pieces. The DNA-protein complexes are then immunoprecipitated from the solution using an antibody which has a high affinity for the protein of interest, and the DNA bound to this protein is then purified, amplified and sequenced, yielding millions of sequencing reads that originate from locations in the genome where the protein of interest was bound. The reads are then mapped to the genome, and regions with a statistically significant enrichment of reads are identified (Zhang et al., 2008). These regions can then be confidently assumed to interact with the protein of interest.

2.3.2 DNA methylation: whole-genome bisulfite sequencing

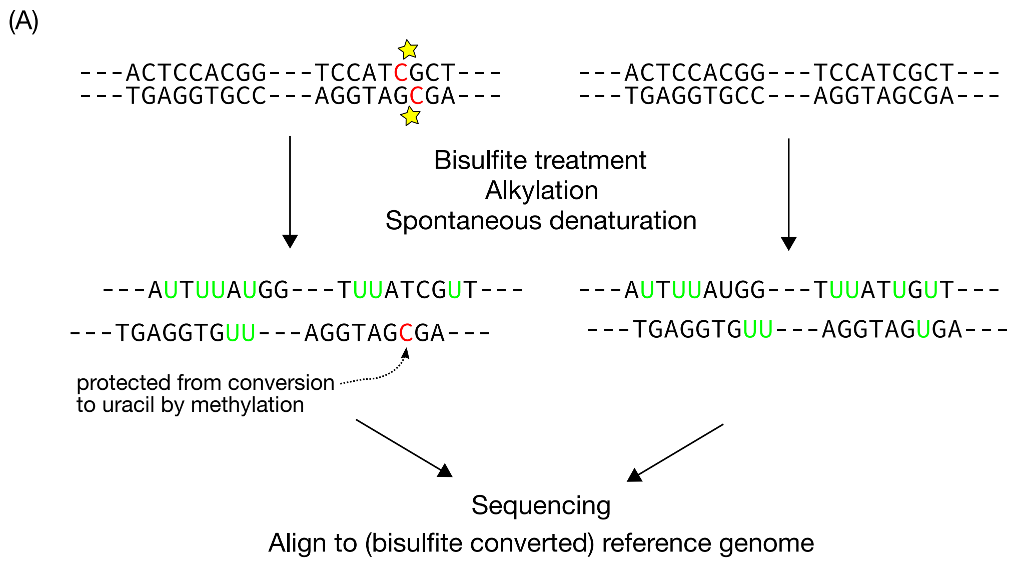
Whole-genome bisulfite sequencing (WGBS) is an experimental method which, given a sample of cells, can determine for each cytosine in the genome the proportion of cells that have that position methylated (Lister et al., 2008; Cokus et al., 2008).

The method (see Figure 2.2A) involves first isolating DNA from the sample of cells, and then bisulfite treating the DNA. This treatment converts non-methylated cytosines to uracils, but leaves methylated cytosines unchanged. The treated DNA is then sequenced, and by comparing the resulting sequence to a reference genome it is possible to determine whether each cytosine in the genome is methylated or not. Each location in the genome is usually sequenced more than once, so for most CpGs in the genome we are able to calculate the proportion of cells in the sample that are methylated at that exact position. This proportion is commonly referred to as a beta (β) value (Figure 2.2B).

One disadvantage of whole-genome bisulfite sequencing is that it requires a huge amount of sequencing, which is an impediment because of the high cost that is involved. This is because the experiment essentially requires resequencing the entire genome at a relatively high depth of coverage. As mentioned above, the vast majority of the genome is constitutively methylated, and contains rare stretches of non-methylated CpGs. If one is interested in studying non-methylated regions of the genome, which make up roughly 5% of the entire genome, it is unnecessary to sequence the other 95% of the genome. Variants of WGBS have been developed to partially alleviate this problem, such as reduced representation bisulfite sequencing (Meissner et al., 2005).

2.3.3 DNA methylation: Bio-CAP

In contrast to whole genome bisulfite sequencing, *Bio-CAP* is an experimental method for specifically profiling clusters of non-methylated CpG dinucleotides genome-wide (Blackledge et al., 2012). The method, when combined with a computational peak calling method such as MACS (Zhang et al., 2008), is able to


(A)
---ACTCCACGG---TCCATCGCT---
---TGAGGTGCC---AGGTAGCGA---
---ACTCCACGG---TCCATCGCT---
---TGAGGTGCC---AGGTAGCGA---
Bisulfite treatment
Alkylation
Spontaneous denaturation
---AUTUUAUGG---TUUATCGUT---
---TGAGGTGUU---AGGTAGCGA---
---AUTUUAUGG---TUUATUGUT---
---TGAGGTGUU---AGGTAGUGA---
protected from conversion to uracil by methylation
Sequencing
Align to (bisulfite converted) reference genome

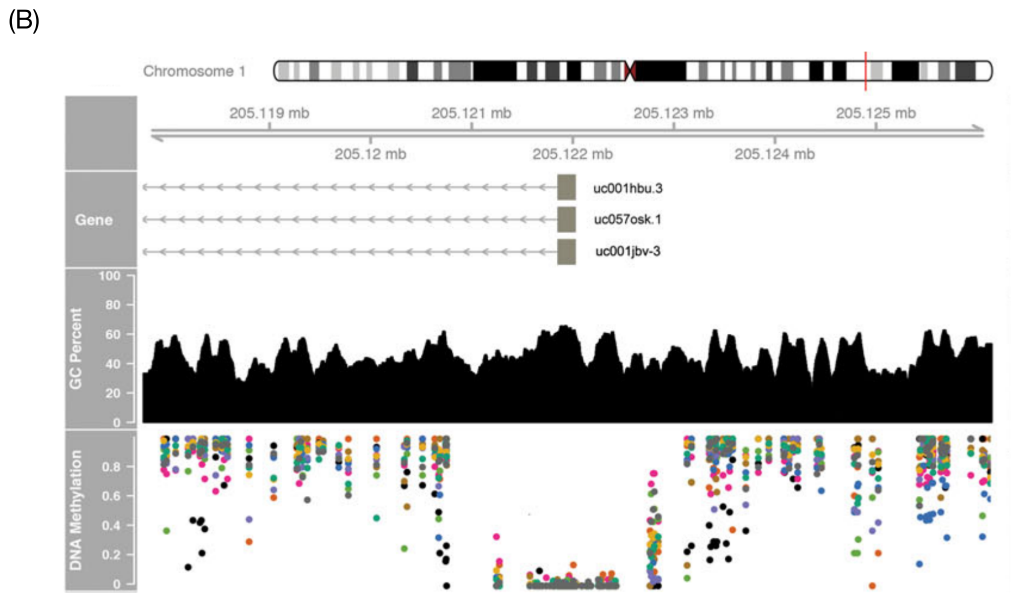


Figure 2.2: Whole-genome bisulfite sequencing experiment. (A) DNA is treated with bisulfite, which causes an unmethylated cytosine to convert to a uracil but leaves methylated cytosines unchanged. These converted sequences are then sequenced using high throughput sequencing, and these bisulfite-generated polymorphisms can be identified by comparison to a reference genome. Adapted from Wikimedia Commons (2016b). (B) After mapping reads and post processing, the results provide a per-CpG estimate of the fraction of cells which are methylated at that position (bottom track). From Mendizabal and Yi (2015).

discriminate non-methylated stretches of CpGs from methylated CpGs, and the signal that is achieved is proportional to the density of non-methylated CpGs in the region. Similar to ChIP-seq experiments (see Section 2.3.1), Bio-CAP is an affinity purification-based method, where the affinity between two biomolecules (e.g. an antibody and protein) is used to enrich a pool of DNA for the regions of interest. In contrast to ChIP-seq, which relies on the affinity between specific pairs of antibodies and proteins, Bio-CAP uses the affinity of a specific protein domain, called a "CxxC zinc finger domain", for unmethylated DNA. This affinity is used to purify unmethylated DNA regions from a pool containing both methylated and unmethylated genomic regions.

In more detail, the experiment begins by attaching CxxC protein domains to a bead which can later be used to isolate whatever is bound to the CxxC domain. The complex is then mixed together with genomic DNA that has been sonicated to approximately 150-250 bp so that the complexes of the CxxC and unmethylated CpG-rich DNA can form (see Figure 2.3A). The mixture is then centrifuged, in order to remove the unbound DNA from the mixture. The remaining DNA (bound to a CxxC-domain) is then washed to further remove any unbound DNA, and then subjected to elution at a high concentration of salt to try to remove proteins and DNA that are binding non-specifically. The high-salt fraction, containing the non-methylated CpG DNA, is then collected and sequenced. These sequences are then mapped to a reference genome, and clusters of reads accumulate at regions of the genome where clusters of non-methylated CpGs are present. A peak calling method is then used to identify these clusters genome-wide. When using MACS (Zhang et al., 2008) for calling peaks, the peak caller adapts its model to the experimental data and to local differences in the genome, and as such is able to provide more reliable discrimination of significant sequencing peaks (non-methylated regions) from noise (methylated regions) than would be possible with a simple cutoff on the number of sequencing reads. The regions identified by the peak caller are the final experimentally determined non-methylated genomic regions (Figure 2.3B).

By purifying and sequencing only the non-methylated regions of the genome, the amount of sequencing that is required is greatly reduced in comparison to WGBS. For example a typical whole genome bisulfite sequencing experiment on a human sample requires between 600 million and 6 billion reads (the examples used were NIH Roadmap Dataset ID ENCFF887XUB and ENCODE Dataset ID ENCSR890UQO), whereas a typical Bio-CAP experiment requires only approximately 60 million reads (Long et al., 2013), a reduction of at least 10 fold. However, there is one clear disadvantage of Bio-CAP compared to WGBS: Bio-CAP does not have base-pair resolution, but rather identifies broad regions of low methylation.

2.4 Other relevant experimental methods

In addition to the measurement of different epigenetic marks, there are several other experiments that are very helpful when studying regulatory regions like enhancers.

2.4.1 CAGE

CAGE, or cap-analysis of gene expression (Kodzius et al., 2006), is a technique for identifying the transcription start sites of RNA transcripts. This can be used when trying to identify enhancers, because some enhancers are thought to generate bidirectional transcripts called eRNAs. Using CAGE, it is possible to identify the transcription start sites of actively transcribed regions genome-wide, and by applying filters based on genome annotation as well as the type of transcription (single direction or bidirectional) one can estimate the activity of protein coding genes, non-coding RNAs, or putative enhancers.

CAGE takes advantage of the fact that the 5' end of some RNA transcripts, which is the end adjacent to the transcription start site, have a modified chemical cap added to them. This cap is used by the cell's translational machinery identify mRNAs that they will translate into a protein. CAGE works by first isolating RNA with a 5' cap, synthesizing a complementary DNA strand, releasing the single stranded DNA and attaching it to an adapter which is used to amplify the 5' sequence so that it can sequenced. After mapping these reads to a reference genome, the reads stack up at the gene's transcription start site in proportion to the amount of capped RNA transcript that was present in the original RNA isolate. These reads can therefore be used as a digital readout of the activity of transcribed genes, and to identify tissue-specific transcription start sites.

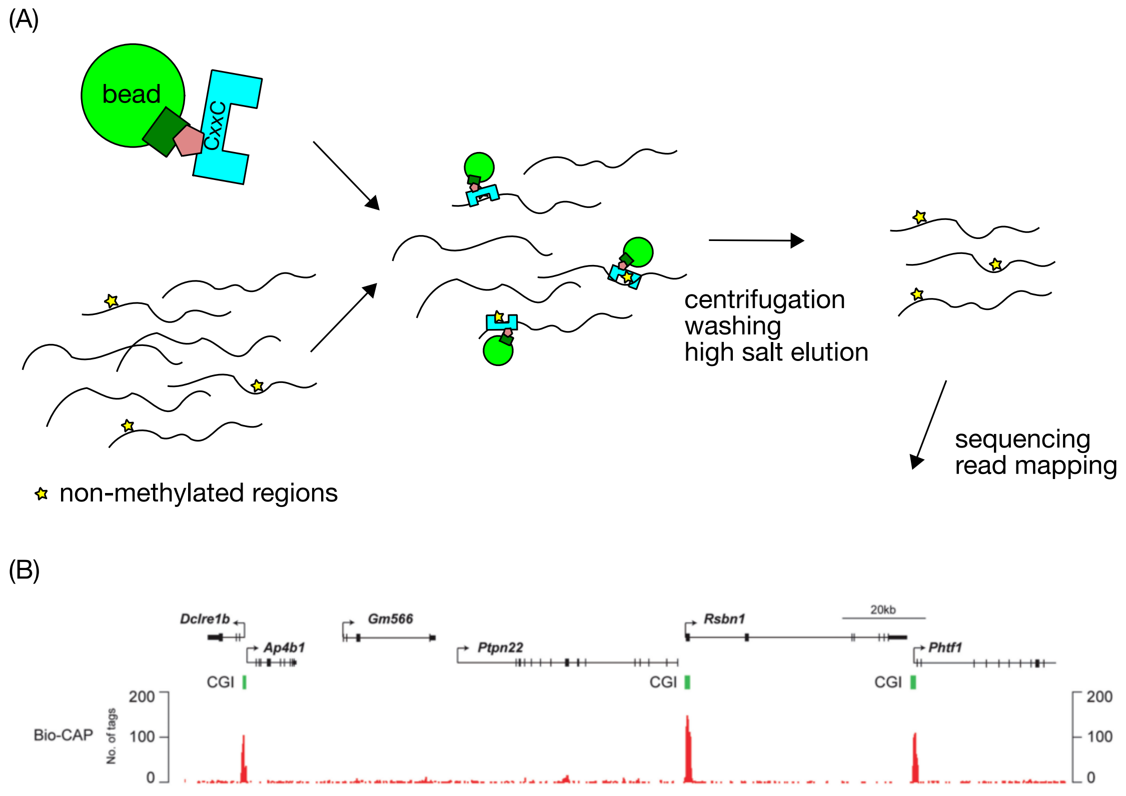


Figure 2.3: Bio-CAP experiment. (A) CxxC domains bound to beads are combined with sonicated DNA. This mixture is then subjected to centrifugation, washing, and elution at high salt concentrations in order to isolate the non-methylated DNA which binds to the CxxC domain. The DNA which remains bound to the beads through this process is then sequenced. (B) An example genomic region showing mapped reads from a Bio-CAP experiment stacking up at regions with clusters of non-methylated CpGs. Adapted from Blackledge et al. (2012).

2.4.2 HiCAP

HiCAP (Sahlén et al., 2015) is a method for identifying DNA-DNA interactions, or regions of the genome that loop together. Specifically, HiCAP enriches for interactions where one of the regions is a promoter, by designing probes that attempt to capture all known promoter regions.

The HiCAP experiment starts by cross-linking the cells, which causes DNA regions that are very close to each other to bind together tightly. The cells are lysed and the nuclei are then isolated, and treated with a restriction enzyme that cuts the DNA into many smaller pieces. The resulting DNA fragments are then treated with biotin, and the entire mixture is then diluted and then the cross-linking is removed, and the fragments that were close to each other will then hopefully ligate to each other, forming a circular DNA consisting of a short region from each interacting DNA fragment. These ligated DNAs are then purified and the promoter probes are used to enrich the resulting sequences for DNAs where at least one of the interacting partners is a promoter. The resulting promoter enriched sequences are then sequenced using paired end sequencing, and the sequences are then carefully mapped to the reference genome so that the two DNA regions that are interacting can be identified.

2.4.3 Enhancer reporter assays

Enhancer reporter assays are used to test if a given piece of DNA can act as an enhancer. Unlike all the previously mentioned methods, the basic enhancer reporter assay is not a genome-wide experiment. Rather, each region of interest has to be tested one at a time. Fortunately there are large databases of regions that have already been tested (see Vista enhancer browser).

The assay works by taking a short stretch of DNA, containing the putative enhancer region, and placing it upstream of a promoter and a reporter gene. This construct, called the reporter construct, is then injected into a tissue or cell type of interest and the activity of the reporter gene is measured. The promoter is usually chosen so that it does not drive expression on its own in the cell type where the enhancer is being tested, so that the activity of the reporter can be attributed to the putative enhancer region. It is also possible to inject the reporter construct into a developing embryo, and then the tissue-specific enhancer activity can be visualized using microscopy (Visel et al., 2007).

An experiment that can be considered a genome-scale version of this assay has been developed, called STARR-seq (Arnold et al., 2013), but not many data sets are currently available and there is some difficulty scaling up the experiment to work on genomes the size the human or mouse genome. Briefly, STARR-seq works by randomly shearing genomic DNA, and then inserting these random sequences into reporter constructs to create a whole library of reporters. These constructs are then injected into cells, and the reporters are designed so that the random DNA regions that are able to act as enhancers drive their own expression, and these sequences can then be isolated and sequenced. When the resulting sequences are mapped back to the genome, we have an enrichment of reads in regions that have functioned as enhancers, and the number of reads is roughly correlated to the strength of their enhancer activity.

Chapter 3

Machine learning on biological sequences

Given the critical role of the genome to living organisms, it is not surprising that a large amount of effort has been put into trying to decode the information it contains. Despite this effort, we are still quite far from being able to predict the phenotypic influence of any given region of the genome.

Fortunately, if we consider the genome as a sequence of text written in an alphabet of four characters, the four nucleotides, then we can leverage the extensive research that has gone into the study of machine learning, natural language processing and sequential data analysis in general. These methods can be used to help us make sense of the genome, by identifying patterns in its sequence which are associated to regions with different functional roles. These patterns can then ideally be used to provide researchers with some initial hypotheses when studying the mechanistic or causative relationships between genomic sequence and phenotypes.

This chapter provides some background on core concepts in machine learning, highlights some methods and applications of machine learning on biological sequences, and gives a more detailed description of the main methods used throughout this thesis.

3.1 Fundamental machine learning concepts

Before we can discuss sequence classification in any detail, we first need to describe what machine learning is and some of its basic concepts.

3.1.1 An example of applying machine learning to a biological problem

Machine learning is a field combining concepts from computer science, statistics, and engineering. The goal of machine learning is to identify patterns in data, and in many cases to use those patterns in order to take actions such as classifying data into different categories (referred to as *classification*) or to predict a numerical value (referred to as *regression*) (Bishop, 2006).

To introduce some important terminology, let us consider an example of applying classification to a biological problem. Say we are given a set of genomic regions and we want to know if they are introns or exons. To approach this problem using machine learning, we could first collect a set of regions where we know the class, or *label*, of the regions: intron or exon. This set of data is called a *training set*, and we will use it to try to identify patterns that can be used to classify our new genomic regions as introns or exons. Based on our biological knowledge, we have some idea that trinucleotide frequencies might differ between these two classes of regions, because the exons consist of trinucleotides called codons that encode for the amino acids that will make up a protein, while the introns do not code for amino acids. Therefore, we look up the DNA sequence of all of our training set regions, and calculate the trinucleotide frequencies of each sequence. This step is called *feature extraction*, and the resulting frequencies are called *features*. Next, we want to use this labeled training data as the input to a machine learning algorithm, which optimizes the parameters of a function, or *model*, to minimize the error when using the function to transform input features to output labels. In the case of our example, the model takes as input the trinucleotide frequencies of any genomic region, and the model will output a prediction of whether it is an intron or exon. This optimization can be called the *training step*. There are many methods which can be used for this purpose, and we will discuss some of them later in this chapter (see the section Methods for machine learning on sequences). Finally, we want to have some idea how well our model works on new data which was not included in our training set. This concept is called *generalization*, and we usually evaluate it by setting aside part of the labeled data from the very beginning and not using it in our training set, so that we can evaluate how well our predictions work on new data after the model has been trained. This set of data is referred to as the *test set*, since we know which label each item in this set is supposed to have and this knowledge can be used to see how good our model's predictions are.

3.1.2 Supervised, unsupervised and semi-supervised learning

There are three broad categories of machine learning tasks that any problem usually falls into: *supervised learning*, *unsupervised learning* and *semi-supervised learning*. These three categories differ based on how much labeled data is incorporated into the training step of the algorithm. For example, a dataset might consist of a set of windows of the genome, the predictive features are the DNA sequences of those regions, and the labels would be if those regions are methylated or if they are instead part of a non-methylated region. Using this labeled data as input, we can use a supervised machine learning algorithm to learn a function which can map new data to its most likely label.

In contrast, with unsupervised learning we have a set of data with predictive features for each data point, but the data is not labeled. Therefore, instead of learning from the example labeled data, we rather try to identify structure (e.g. clusters) in the unlabeled dataset. In comparison to the previous example, this would be like having a dataset of windows of the genome, along with their DNA sequences, but no labels indicating whether the regions are methylated or not. In this case we can only look for patterns in the DNA sequences that could be used to group them together in some way, but we wouldn't necessarily know the biological meaning of those groups without considering additional information which was not included in the analysis, such as genomic annotation of genes, non-coding RNAs, repetitive sequences and CpG islands. In the case of clustering, the biological relevance (if any) of the clusters that are identified would need to be determined after the clustering has been performed, often by looking at examples from each of the groups manually as well as the features that contribute most to the clustering.

Semi-supervised learning is a compromise between supervised and unsupervised learning. With semi-supervised learning, some of our data is labeled and some of it is unlabeled. These methods take advantage of an assumption that the underlying function that we are trying to model is “smooth”, so we can say that similar input data is expected to have similar labels. Therefore, we can use the unlabeled data to define high density areas of the input space (which would be assumed to share a label based on the smoothness assumption), and then apply a label to these regions based on the few labeled data points that we have.

There are several reasons why one might choose to use supervised, unsupervised or semi-supervised learning. For example, it can often be very costly to determine labels, for instance when a biological experiment is very time-consuming or expensive. In this case, one would be forced to use unsupervised methods if labeled data is not available, or semi-supervised methods if there are at least a small amount of labeled data. Additionally, with supervised methods you are only able to choose between pre-determined labels when classifying a data point. This is not desirable when doing exploratory analysis so you might choose to use unsupervised learning first. In contrast, when the set of possible outcomes or labels is already known, and there is a large amount of labeled data available, supervised learning methods will tend to be better than unsupervised learning methods at predicting the label of new data, as well as identifying important features that are useful for differentiating between members of each class.

Semi-supervised meta-algorithms: self-training and co-training Two of the simplest semi-supervised learning methods are *self-training* and *co-training*. They can be considered meta-algorithms, because they use existing classification algorithms (see Section 3.2 for some examples), along with unlabeled data, to try to iteratively increase classification performance (Aggarwal, 2015). Both methods start with a (small) labeled set L, a (large) unlabeled set U, the size of the subset from U that should be considered at each iteration n_u, and some parameter k which indicates how many data points should be moved from the unlabeled set to the labeled set with each iteration of the algorithm.

- **Self-training** is the simpler of the two methods, and it involves training a single classifier on L, using that trained classifier to identify the set K of the k most confident instances in a random subset of size n_u from U, U_1. Those instances K are then moved from U to L ($U \leftarrow U \setminus K, L \leftarrow L \cup K$), and the label predicted by the classifier is assigned to the instances. This process is repeated until some stopping criterion is reached, for instance L is increased to a certain size, or all of the data in U has been labeled.

- **Co-training** is a more complex method than self-training, and involves using two feature sets X_1 and X_2 to describe the data (Blum and Mitchell, 1998). Two separate classifiers g_1 and g_2 are trained on L, using the features X_1 and X_2 respectively. Then each classifier independently identifies the k most confident instances in two random subsets of size n_u from U, U_1 and U_2. These instances K_1 and K_2 are then moved from U to L if the two classifiers do not predict conflicting labels. As above, the process is repeated until some stopping criteria is reached or all data in U is labeled. For more detail, see Algorithm 1.

  Augmenting training data in this way has led to numerous successful applications, for instance in text and website category classification (Blum and Mitchell, 1998), prediction of geographical location (Riloff and Jones, 1999), word sense disambiguation (Yarowsky, 1995) and named entity classification (Collins and Singer, 1999). A successful bioinformatics application uses co-training to improve disease phenotype prediction from genotype by using a second classifier to impute the phenotype of unlabeled patients based on a second class of information: clinical health records (Roqueiro et al., 2015). In general, a review study has shown that algorithms that make use of an independent split of the features outperform algorithms that do not (Nigam and Ghani, 2000).

3.1.3 Supervised learning with different numbers of classes

When performing supervised learning, our goal is to take some input vector and assign it to one or more out of K classes. The most common setting in supervised learning is *binary classification*, where $K = 2$, meaning that each input vector needs to be mapped to one of two possible classes. One example that is explored later in this thesis is predicting whether a given genomic region is an enhancer

Algorithm 1 Co-training

```
Input: high-confidence labeled set $L$, unlabeled set $U$, parameter vector $\theta = (n_u, k)$
Output: classifiers $g_1$ and $g_2$ trained on augmented labeled set $L$
 1: while stopping criterion is not met do
 2:     for $i = 1, 2$ do
 3:         train classifier $g_i$ on $L$
 4:         sample subset $U_i$ of size $n_u$ from $U$
 5:         predict class probabilities of data in $U_i$ with $g_i$
 6:         determine the $k$ most confidently predicted data in $U_i$
 7:         label the $k$ most confidently predicted data in $U_i$
 8:     end for
 9:     if intersection of $U_1$ and $U_2$ is not empty $(U_1 \cap U_2 \neq \emptyset)$ then
10:         discard labeled data with non-matching labels $(g_1(x_j) \neq g_2(x_j))$
11:     end if
12:     add labeled data from $U_1$ and $U_2$ to $L$
13: end while
```

(class 1) or not (class 2). We can define this mathematically as:

$$Y = f(X), \tag{3.1}$$

where Y is a vector of length n consisting of class labels C_k, $k = 1, 2$, X is an n-by-m matrix of features that describe the input data, and $f(X)$ is a function mapping the features to one of the two classes C_1 or C_2.

In contrast, *multiclass learning* is an extension of this idea where $K > 2$. It is defined identically to the above, but the vector Y consists of class labels C_k, $k = 1, 2, 3, ...$, meaning that each item can be assigned to one of three or more mutually exclusive classes. An example of this type of problem in biology would be predicting the stage of a cancer sample given the set of regions of the genome that have variable DNA methylation patterns. In that situation there are more than two possible stages, and each stage is mutually exclusive.

A variation on binary and multiclass classification is *multilabel classification.* This differs from binary and multiclass learning because each input item can be assigned to multiple classes simultaneously. For example, when predicting the set of tissues in which a gene is expressed given the DNA sequence of the gene's promoter region. Unlike the multiclass example, a gene can be active in more than one tissue simultaneously, so the output classes are not mutually exclusive. In this case, our variable Y is now an n-by-K matrix where n is the number of input items, each of the K columns contains a binary value indicating whether or not the item belongs to that class, and $f(X)$ needs to map the input features to a set of classes rather than a single class. This is an example of multilabel classification with binary classifiers (true or false for each tissue), but it is also possible to do multilabel multiclass classification where each label is selected from more than two classes.

3.1.4 Performance Metrics

In order to evaluate the performance of a classifier we need to define performance metrics. In situations where the predictions assign a class to each element being classified, the metrics that we use in this work are: *true positive rate* (also known as *recall*), *false positive rate*, and *precision*.

To define these measures, we first need to define true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). They are easily explained using the following confusion matrix showing the relationship between the predicted class versus the actual class of each item:

		Predicted Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

We can now define our performance metrics as follows:

$$\begin{aligned} \text{true positive rate} &= TP/(FN + TP) \\ \text{false positive rate} &= FP/(FP + TN) \\ \text{precision} &= TP/(TP + FP) \end{aligned}$$

Additionally, in situations where the classifier assigns a score indicating the confidence that each element belongs to a given class we use two additional per-

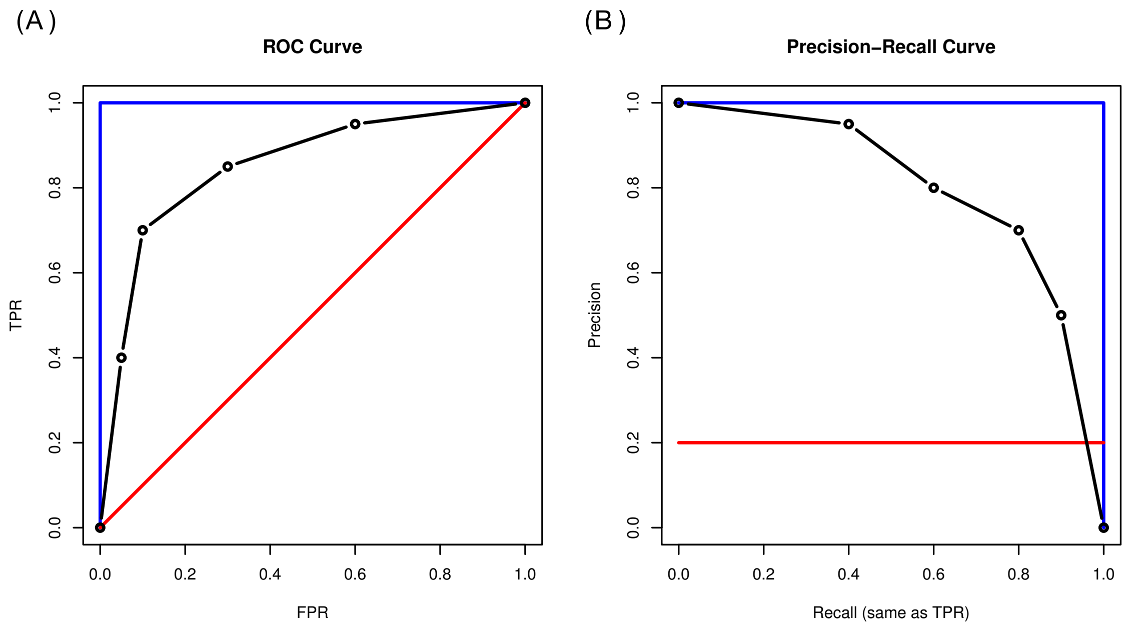


Figure 3.1: Example ROC and Precision-Recall Curves. Plots showing the optimal curve (blue), curve for random predictions (red), and a typical curve for a relatively good classifier (black) for a receiver operating characteristic curve (A) and precision-recall curve (B). The line showing expected performance for random predictions in the precision-recall curve changes depending on the relative abundance of the two classes being predicted, and the precision is expected to reach the fraction of positive/negative examples, in this case 1/5 or 0.2.

formance measures: the area under the receiver operating characteristic curve (AUROC) and the area under the precision-recall curve (AUPRC).

Receiver operating characteristic (ROC) curves are a commonly used plot for showing the performance of a binary classifier as the classifier's score threshold is varied (Figure 3.1A). At each value of the score threshold the true positive rate (TPR) and false positive rate (FPR) is calculated. Plotting a line connecting these pairs (TPR, FPR) in the order of increasing score threshold gives the ROC curve of the classifier. The area under this curve is then calculated and can be interpreted as the probability that a randomly selected element from the positive class will score higher than a randomly selected element from the negative class. Values near 0.5 indicate that the classifier performs similarly to randomly selecting a class, and an AUROC of 1.0 is a perfect classifier.

While ROC curves are the most common performance measure for evaluating classifiers that output a confidence level, we also use precision-recall curves to evaluate many of the classifiers. This is because ROC curves are known to be insensitive to class imbalance (Saito and Rehmsmeier, 2015), and the genome-wide predictions that we perform in parts of this thesis are always leading to a highly imbalanced class distribution. For example, non-methylated islands only make up roughly 2-4% of each organism's genome, so when trying to predict them throughout the entire genome we have a ratio of negative:positive regions of roughly 30:1. This is a difficult situation because a classifier that appears to perform well in the balanced setting might be nearly useless in the unbalanced setting, because it is very hard to control the number of false positives in such a situation. To give a better idea of how practical these methods are for the genome-wide prediction it is better to look at precision-recall curves, which plot the fraction of predicted positive regions correctly identified (precision) versus the fraction of all positive regions that are correctly predicted (recall) as the score cutoff used to assign each region to a class is varied across all possible values (Figure 3.1B). This gives us an idea of the false discovery rate (1 - precision) of the method at different score thresholds and whether the genome-wide predictions the classifier makes are likely to contain many false positives.

3.1.5 Cross-validation

When training a classifier, the performance on the training set is not a good estimate of how well the classifier will perform on new data. This is because most classifiers are prone to overfitting, which means that they achieve very low prediction error on the data they were trained on, but a much higher error on new data, because they overspecialize the model to the training data and learn to reproduce the noise within that data. In order to overcome this problem, one approach is to set aside a portion of the data which is not used in the training step and use it as an independent dataset or test set in order to assess how well the classifier generalizes to new data. An extension to the concept of using a separate test set is to split the whole dataset into m non-overlapping sets (commonly called “folds”), and iteratively use each set as a test set while using the remaining $m-1$ sets as training data. This is called m-fold cross-validation. It is preferred in many situations because it allows us to use all the data when building the model, while at the same time giving a good estimate of predictive performance. The maximum number of folds is when m is equal to the number of elements in the data set, which leads to each set consisting of a single element. This special case is referred to as “leave-one-out” cross-validation.

One downside of cross-validation is that the total number of training runs increases by a factor of m. However, because each run is independent we can often overcome this issue by executing each run in parallel.

3.1.6 Feature importance

In contrast to some other situations where machine learning may be applied, it is especially important when analyzing a biological problem to not only find a solution which gives an objectively good performance, but it is additionally important to improve our biological understanding of the problem. To this end, we are particularly interested in the features of the input data points that were most useful when assigning the data to a label, commonly referred to as feature importance. It is similar but not identical to the concept of feature selection, which aims to remove redundant or irrelevant features from the model.

3.2 Methods for machine learning on sequences

The following sections are a non-exhaustive overview of machine learning methods on sequences that have been applied to biological data.

3.2.1 Linear models

Linear models are a large class of supervised learning methods for both regression (when the target variable is numerical) and classification (when the target variable is categorical). These methods take an object described using some numeric input variables x, and learn a function that maps x to a target variable y, by tuning a set of weights w for each of values in x. The models are linear because there is a linear relationship between the output variable y and the weights w. The simplest example of a linear model is *linear regression*, which assumes that the target variable y is a linear combination of the input variables x scaled by the weights w (see Bishop, 2006 for an extensive overview):

$$y(x, w) = w_0 + w_1x_1 + ... + w_Dx_D \tag{3.2}$$

It is also possible to make this concept more general by allowing a non-linear function ϕ, called a basis function, to be used to transform the input variables. This allows non-linear relationships between the input features and the target variable, but still requires a linear relationship between the weights and the target variable:

$$y(x, w) = w_0 + \sum_{j=1}^{D} w_j\phi_j(x), \tag{3.3}$$

where $y \in \mathbb{R}$.

In order to fit the model, or solve for w, we need a loss function L which measures how "bad" our predictions are in an objective sense. With regression, we commonly try to minimize the sum of the squared errors over all N of our predictions:

$$L^{ls}(w) = \sum_{i=1}^{N} \big(y_i - f(x_i)\big)^2 \tag{3.4}$$

A further variant of this, called *logistic regression*, transforms the linear function of the feature vector with the sigmoid logistic function σ:

$$y(x, w) = \sigma\big(w_0 + \sum_{j=1}^{D} w_j\phi_j(x)\big), \tag{3.5}$$

where $y \in [0, 1]$.

Despite its name, logistic regression is a model for classification, and in a binary classification setting the resulting outputs of the model can be considered as posterior probabilities of the item belonging to the positive class. We also use a different loss function for logistic regression, in order to find the parameters w

which maximize the sum of the posterior probabilities $p(y_i|x_i)$ (the probability assigned to the correct class given the input data):

$$L^{log-likelihood}(w) = \sum_{i=1}^{N} \log p(y_i|x_i) \tag{3.6}$$

When classifying strings, there is one additional step that needs to be performed that we have ignored in the previous descriptions. We have mentioned that the input variables x need to be numeric, however when working with DNA sequences the input is actually a string. In order to use strings as input variables, we need to explicitly convert the strings to numeric values. There are various conversions that are possible depending on the application, with the most common being the conversion of the string to a vector of substring frequencies, or by using a set of known TF binding motifs and counting the number of binding sites or strength of predicted binding of these TFs to the input DNA sequence. It should be noted that when using long substring frequencies, this explicit vector of counts can be quite large, which is one potential downside of this method. We will look at one way to address this problem in the next section which describes support vector machines and kernels.

Feature importance There are several methods for identifying important features when using linear models. The first set of methods are *stepwise methods*, in which features are added to or removed from the model iteratively. The most common approaches are *forward selection*, where features are iteratively added to the model, *backward elimination*, where features are removed from a model that initially contains all features, and *bidirectional elimination*, where features can be either removed or added to the model at each step. The general concept is to consider the addition or removal of each feature from the model at each step, usually using a statistical test such as an F-test, or some other measure such as the Bayesian Information Criterion, Akaike information criterion, false discovery rate, or adjusted R^2 (Hocking, 1976; Kundu and Murali, 1996). However, there are several disadvantages of stepwise methods. The iterative model fitting is computationally intensive, the results can be overly biased and optimistic, and the methods perform poorly in the presence of correlated input variables (Harrell, 2001).

An alternative to stepwise methods is to use *regularization*, which adds an extra term to the loss function which increases the error when the weights used in the model are large. Unlike the previous methods, this is an "embedded method" because the feature importance is directly included in the model fitting procedure. The purpose of regularization is to reduce overfitting and make the model more generalizable, but it can also be used to make the model more sparse, meaning that some coefficients are set to zero and those features are essentially removed from the model. For example if we consider the *sum-of-squares error* from the previous section, a general regularized version would be:

$$\underbrace{\sum_{j=1}^{M-1}(y_j - f(x_j))^2 +}_{\text{data dependent error}} \underbrace{\frac{\lambda}{2}\sum_{i=1}^{N}|w_i|^q}_{\text{regularization term}} \quad , \tag{3.7}$$

where λ is a coefficient which determines how much of an influence the regularization term takes in comparison to the error term, and q determines the contour of the regularization term. Setting $q = 1$ gives you the so-called *lasso* (Tibshirani, 1996), which is a regularization term that is well suited for creating a sparse model. When using the lasso regularization term, increasing λ will lead to more coefficients being set to zero and a more sparse model, and this parameter is usually set using cross-validation.

Biological applications One of the early examples of a string-based regression method that was applied to a biological problem was the REDUCE algorithm of Bussemaker, Li, and Siggia (2001). The goal of the method was to identify cis-regulatory elements, particularly those associated with known transcription factor binding sites, in the promoter sequence of genes that have an influence on gene expression. This method is based on the following simple linear model:

$$A_g = C + \sum_{\mu \in M} F_\mu N_{\mu G} \tag{3.8}$$

where A_g is the log transformed ratio of mRNA abundances between two samples for gene g, C is the expression level without any motif instances in a region, $N_{\mu g}$ is the number of occurrences of motif μ in the regulatory region of gene g, and F_μ is the change in expression per instance of motif μ. The coefficients F_μ were determined using least squares regression (i.e. linear regression using the least-squares loss function). The "motifs" that were considered included all DNA subsequences up to 7 nucleotides in length, which were iteratively added to the model by selecting the single motif which results in the greatest reduction in error when added to the model. The authors decided when to stop adding more motifs to the model by evaluating the statistical significance of the change in error that is achieved by adding a new motif to the model, under the assumption that distribution of changes in error follow a normal distribution, and placing a cutoff of 0.01 on the resulting p-value.

This basic concept was subsequently expanded upon by Conlon et al. (2003) by using motif match scores instead of binary motif hits, Keleş, Laan, and Vulpe (2004) using logic regression, and Das, Banerjee, and Zhang (2004) using multivariate spline models (Friedman, 1991). People consequently moved away from predicting expression based on sequence data and instead focused on predicting expression using experimentally determined TF binding data (Gao, Foat, and Bussemaker, 2004) or histone modification levels (Karlić et al., 2010).

3.2.2 Support vector machines

A support vector machine (SVM, Cortes and Vapnik, 1995) is a generalization of the *optimal hyperplane* algorithm, which seeks to find an optimal boundary (or hyperplane) between two sets of data: positive and negative training data. This hyperplane can then be used to classify new examples x':

$$f(x') = \operatorname{sign}(wx' + b), \tag{3.9}$$

where b is a bias term.

It does this by identifying the maximum margin between the data points belonging to each class (Hastie, Tibshirani, and Friedman, 2009). SVMs differ from simple optimal hyperplane classifiers however, in that they do not require the data from each class to be separable, which means that there does not need to exist a (hyper)plane that can separate the two classes of data points so that all the points from one class are on one side of the plane, and the points of the other class are on the other side. This is critical because when working with real-world data we rarely have data that is separable. We therefore have an optimization problem which tries to compromise between finding the largest margin while softly penalizing points that lie on the wrong side of the classification margin boundary, by minimizing:

$$L^{\text{SVM}}(w, \xi) = \underbrace{C \sum_{n=1}^{N} \xi_n}_{\text{misclassification penalty}} + \underbrace{\frac{1}{2} \|w\|^2}_{\text{margin width}} , \tag{3.10}$$

where $C > 0$ controls the tradeoff between the two goals. Misclassified points n are weighted using slack variables ξ_n according to their distance from the correct side of the margin (see Figure 3.2).

As with the *basis functions* we used with linear models (see previous section), it is possible to transform features from the input space to some other feature space using a feature mapping $\Phi(x)$. Additionally, an SVM is a *kernel method*, which means that the algorithm does not require that each data point has its own numerical representation, but rather it is only necessary to be able to calculate pairwise similarities between data points (Schölkopf, Tsuda, and Vert, 2004). This is because in the SVM decision function (see Equation 3.9) can be rewritten in the so-called dual form, so that the value of the decision function for a new data point x' only uses the dot products between the new point and a subset of the training data (called the "support vectors") in the feature space $\Phi(x_i) \cdot \Phi(x')$:

$$f(x') = \text{sign}\Big(\sum_{\text{support vectors } x_i} \alpha_i y_i (\Phi(x_i) \cdot \Phi(x')) + b \Big) \tag{3.11}$$

and the dot product between these feature vectors can be replaced by a *kernel function* $K(x_i, x') = \Phi(x_i) \cdot \Phi(x')$. By using the kernel function it is no longer necessary to calculate the explicit representation of the data points in the feature space defined by Φ, rather the value of the kernel function can be calculated directly. In order to be a valid kernel function, the function must be symmetric

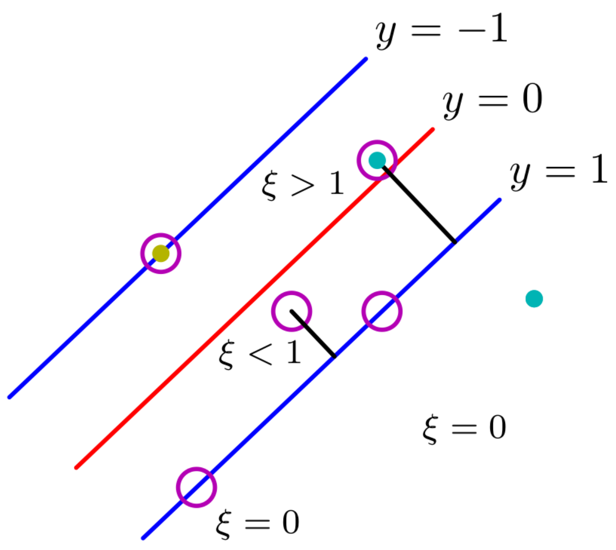


Figure 3.2: Support vector machine. The decision boundary (or separating hyperplane) is shown in red, the support vectors are the points that are indicated with circles, the margin is the region between the two blue lines. Slack variables ξ are >0 for points that are on the incorrect side of their margin boundary, and zero otherwise. y is the decision value, where the sign of y indicates the class that is predicted. From Bishop (2006).

$(K(x, z) = K(z, x))$, and positive definite:

$$\sum_{j,k=1}^{N} c_j c_k K(x_j, x_k) \geq 0 \tag{3.12}$$

for any $N > 0$, and any $c_1, \dots, c_N \in \mathbb{R}$ (Schölkopf, Tsuda, and Vert, 2004).

An advantage of using a kernel method like support vector machines is that kernels allow us to work with objects that are not naturally represented by a numeric vector, which is the expected input of almost all standard machine learning methods. These types of objects are common in biology, where the typical objects that we use as input to our machine learning algorithms are often DNA, RNA or protein sequences, and are naturally represented by a string of characters over an alphabet of nucleotides or amino acids rather than a set of numeric values. By using kernel methods we do not need to worry about how to transform this data into a representation that can be handled by the machine learning method, nor adapt the learning method to work with non-numerical data. Instead, we just need to define a kernel $K : X \times X \to \mathbb{R}$ which can be thought of as a comparison function or similarity measure, where higher values indicate more similar objects. This allows us to leverage the decades of research into different pairwise sequence comparison methods without having to adapt the classification method itself to each data representation.

Position-independent string kernels In this thesis we are interested in using DNA sequences of varying length as input, whose sequence features have no known fixed location within this input sequence. In this setting, the most appropriate string kernels are position-independent string kernels, which generally define similarity based on shared subsequences with no concern for the location of these shared subsequences within the input sequence. Here we will describe three such kernels that are used in the thesis: the spectrum kernel, mismatch kernel, and gappy pair kernel.

- **Spectrum kernel**

  The most simple position-independent string kernel is the k-spectrum kernel (Leslie, Eskin, and Noble, 2002). It is a variant of the classic bag-of-words representation of a written text (Salton, 1989), which represents a document using the unordered set of words it contains. These words are delimited by spaces or punctuation, and the words are stored as a sparse vector of indicator values (1 if the word is present in the text, 0 if not), or word frequencies. The similarity between two texts is then defined as the dot product of these vectors. Because DNA sequence does not have obvious word boundaries in the same way that written text does, the spectrum kernel uses a set of features Σ^k consisting of all k-length strings over the alphabet Σ ($\{$A, C, G, T$\}$ in the case of DNA). These k-length substrings are referred to as k-mers. The feature map for a single k-mer a is defined as:

$$\Phi_k^{\text{Spectrum}}(x) = (\phi_a(x))_{a \in \Sigma^k}, \tag{3.13}$$

where $\phi_a(x) = \#$ occurrences of the k-mer a in x. The spectrum kernel is then defined as the inner product of the spectrum feature vectors:

$$K_k^{\text{Spectrum}}(x, y) = \langle \Phi_k^{\text{Spectrum}}(x), \Phi_k^{\text{Spectrum}}(y) \rangle \tag{3.14}$$

The main advantage of the spectrum kernel compared to other string kernels is the fact that it can be very efficiently computed (see Table 3.1). Using suffix trees (Vishwanathan and Smola, 2003), the runtime for a single element in the kernel matrix is linear on the sum of the lengths of the input sequences ($O(l_x + l_{x'})$), and can be reduced even further in certain cases by preprocessing the input sequences. The disadvantage of the spectrum kernel is that for a k-mer to contribute to the similarity score when comparing a sequence to another, it must have an exact match in the other sequence. For short k-mers this is not a problem, but for longer k-mers the chance of finding an identical sequence elsewhere in the genome drops rapidly to zero. Also, we know from analyzing transcription factor binding sites that it is common to have several bases that are important for binding, interspersed with one or more unimportant bases. In this situation it would be preferable to have a kernel that allows for inexact matching.

- **Mismatch kernel**

The (k, m)-mismatch kernel (Leslie et al., 2004) extends upon the spectrum kernel by allowing some inexact matching between k-mers. This set of inexact matching k-mers is referred to as the (k, m)-neighbourhood of the k-mer a, and the neighbourhood contains all k-length sequences β that differ from a by at most m characters. The neighbourhood is denoted by $N_{(k,m)}(a)$. The feature map for a single k-mer a is defined as:

$$\Phi_{(k,m)}^{\text{Mismatch}}(a) = (\phi_\beta(a))_{\beta \in \Sigma^k}, \tag{3.15}$$

where $\phi_\beta(a) = 1$ if β is in the set $N_{(k,m)}(a)$, and 0 otherwise. Then we can define the feature map over the input string x as:

$$\Phi^{\text{Mismatch}}_{(k,m)}(x) = \sum_{\text{k-mers } a \text{ in } x} \Phi^{\text{Mismatch}}_{(k,m)}(a) \tag{3.16}$$

As with the spectrum kernel, the mismatch kernel can then be defined by taking the inner product of the feature vectors:

$$K^{\text{Mismatch}}_{(k,m)}(x, y) = \langle \Phi^{\text{Mismatch}}_{(k,m)}(x), \Phi^{\text{Mismatch}}_{(k,m)}(y) \rangle \tag{3.17}$$

Note that the spectrum kernel is equivalent to the mismatch kernel with m = 0.

- **Gappy pair kernel**

The (k, g)-gappy pair kernel (Kuksa, Huang, and Pavlovic, 2008; Palme, Hochreiter, and Bodenhofer, 2015) considers two k-mers separated by up to g positions:

$$\underbrace{\text{k-mer}_a}_{k}\,\underbrace{\ldots}_{0 \leq i \leq g}\,\underbrace{\text{k-mer}_b}_{k}$$

This differs from the mismatch kernel because of the variable distance that is permitted between the k-mers, and the sequence contains two k-mers, meaning that the number of matching characters between sequences is $2k$. The set of sequences in the (k, g)-gappy pair of the k-mers a and b consist of all sequences γ of the form acb where c is any sequence with a length between 0 and g. This set of sequences is defined as $G_{(k,g)}(a, b)$. The feature map for the gappy pair kernel is defined as:

$$\Phi^{\text{Gappy pair}}_{(k,g)}(a, b) = (\phi_\gamma(a, b))_{\gamma \in \Sigma^k}, \tag{3.18}$$

where $\phi_\gamma(a, b) = 1$ if γ is in the set $G_{(k,g)}(a, b)$, and 0 otherwise. In the same way as we did for the mismatch kernel, we can define the feature map over

Table 3.1: Runtime complexity of some string kernels. Runtimes are for the computation of the entire kernel matrix. p is the average length of a sequence, m is the number of sequences in the training set, k is the k-mer length, M is the number of allowed mismatches, $|\Sigma|$ is the size of the alphabet (4 for DNA), c_K is a constant that is independent of alphabet size (Schölkopf, Tsuda, and Vert, 2004).

Kernel	Complexity
spectrum	$O(pm^2)$
mismatch	$O(k^M|\Sigma|^M pm^2)$
gappy	$O(c_K pm^2)$

an input string x as:

$$\Phi_{(l,g)}^{\text{Gappy pair}}(x) = \sum_{\text{k-mer pairs } (a,b) \text{ in } x} \Phi_{(k,g)}^{\text{Gappy pair}}(a,b) \tag{3.19}$$

As with the other kernels, the gappy pair kernel is defined as the inner product of the feature vectors:

$$K_{(k,g)}^{\text{Gappy pair}}(x,y) = \langle \Phi_{(k,g)}^{\text{Gappy pair}}(x), \Phi_{(k,g)}^{\text{Gappy pair}}(y) \rangle \tag{3.20}$$

Feature importance In general, calculating feature importance is not straightforward with support vector machines. However, in special cases it is possible to retrieve feature weights using the fitted model. In the case of the position-independent string kernels discussed above, the importance z of each feature (e.g. k-mer when using the spectrum kernel) can be calculated as outlined in the original spectrum kernel paper of Leslie, Eskin, and Noble (2002):

$$z = \sum_{\text{support vectors } x_i} \alpha_i y_i \Phi(x_i), \tag{3.21}$$

where α is the optimal solution to the SVM problem in its dual form (see Equation 3.11), i are the indexes of the support vectors, y_i is the true class of the element (-1 for the negative class or 1 for the positive class), and Φ is the spectrum of k-mers for the sequence x_i. The higher the absolute value z the more that k-mer contributes to the classification of a new sequence.

Biological applications When it comes to sequence based classification in bioinformatics, there is no method that has been used as extensively as support vector machines, and prior to the recent improvements in deep learning the SVM was the state-of-the-art method for many types of problems including those in computational biology. Early biological applications include the detection of remote homology between proteins (Jaakkola, Diekhans, and Haussler, 1999; Logan et al., 2001), the functional classification of genes based on their promoters (Pavlidis et al., 2001), mRNA splice site detection (Degroeve et al., 2002), and many others (see Schölkopf, Tsuda, and Vert, 2004 for more examples).

3.2.3 Deep neural networks

Neural networks have made a huge resurgence in popularity over the last few years due to both advances in computational hardware (allowing extremely fast matrix computation on graphical processing units) and the learning algorithms themselves. These advances have made it feasible for the first time to train large neural networks with multiple interconnected layers of nodes using huge datasets consisting of billions of data points. These networks are called *deep neural networks* because of the multiple internal layers that are used, on contrast to shallow networks consisting of an input layer, output layer, and a single internal layer. The first applications of deep neural networks concentrated on speech recognition (Hinton et al., 2012), image recognition (Krizhevsky, Sutskever, and Hinton, 2012), and the generation and translation of text (Martens, 2011; Sutskever, Vinyals, and Le, 2014), and established these methods as the state of the art in each of those fields.

A neural network can be visualized as a graph with multiple layers, which usually consists of one layer of inputs, one or more "hidden" layers, and an output layer (see Figure 3.3). Each of these layers can consist of any number of nodes, which are usually fully connected to the previous layer and have an associated activation function which transforms the output of the node (like the logistic function in logistic regression). One of the most classic types of neural network is the

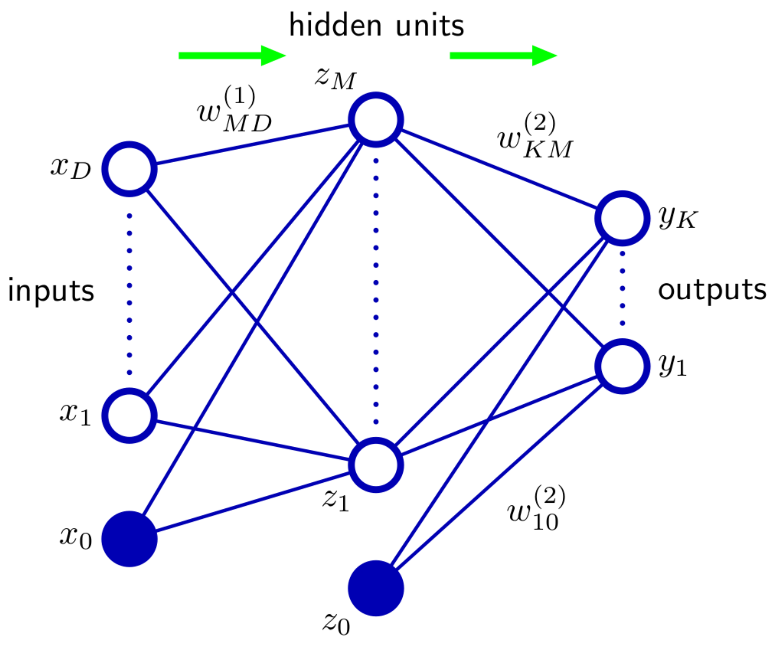


Figure 3.3: A basic feed-forward neural network. The network consists of an input layer $x = (x_0, \dots, x_D)$, a single hidden layer $z = (z_0, \dots, z_M)$, and the output layer $y = (y_1, \dots, y_K)$. Each layer is connected with weights $w^{(l)}_{ij}$ which is the weight of the edges connecting element j in the layer l with element i in layer $l + 1$. The dark blue nodes x_0 and z_0 are biases. Arrows represent the direction of information flow. From Bishop (2006).

feed-forward neural network, also called the multilayer perceptron. It is essentially a model made up of multiple layers of logistic regression models.

Additionally, given the modular nature of neural networks there are many other variants besides feed-forward networks. Especially noteworthy are two types of deep neural networks that are particularly useful for classification based on DNA sequences: *convolutional neural networks* and *recurrent neural networks.*

Convolutional neural networks (CNNs) are appealing because they are designed to require little pre-processing of the data, in contrast to most other methods which require features to be designed by hand. Convolutional neural networks use “filters”, relatively small matrices whose weights are learned during the training process. These filters are then slid over the input sequence, and the element-wise product between the filter and that portion of the sequence is taken. The sum of these values is then computed, and that is the output for the given position in the sequence. This is repeated for the entire input sequence. In this way, the features (filters in this case) are learned at the same time as the network is being trained. For example, several of the methods we have mentioned in the previous sections use a set of known transcription factor binding motifs and score sequences based on the likelihood of each transcription factor binding to that DNA sequence. These scores are then used as input features when training a classifier. Using a CNN, the input of the classifier is the raw DNA sequences, and the motifs are the filters that are learned as the model is fit. This has the advantage of not depending on an extensive set of known TF binding motifs, which could be difficult to obtain especially in species which are not heavily studied.

Recurrent neural networks (RNNs) are well suited to learning from DNA sequence because unlike most other neural networks which assume that each input and output is independent from the others, RNNs are able to take advantage of the fact that the inputs and outputs of the network are sequential. This is accomplished by having the network store some internal state that is used as an additional input when the next input feature is processed, serving as a sort of memory and allowing information to be shared between adjacent inputs or outputs in the sequence.

Feature importance How to calculate feature importance using deep neural networks is still very much an open problem. This difficulty is arguably the greatest disadvantage of using deep neural networks, especially for biological problems. The challenge comes from the high complexity of the network's structure. When using convolutional networks, it is possible to visualize the weights of the filter that is being used. For biological sequences, these filters can be visualized as motifs, and this allows the user to compare these motifs to known TF binding motifs as in Angermueller et al. (2017).

One other relatively recent method by Li, Chen, and Wasserman (2015) simply puts an additional layer between the input layer and first hidden layer with a single weight connecting each node, and uses the weights that are learned for this layer as feature importance measures. This approach has however not gained widespread popularity, possibly because the features can be re-weighted in later layers of the network so the weights at the first layer are not necessarily indicative of the features' contribution to the final prediction.

Biological applications One of the first applications of deep learning to a biological problem was by Leung et al. (2014), where the group tried to predict the alternative splicing of mRNA transcripts across five tissues, using 1393 sequence-derived features including predicted motif binding to the intronic region, exonic region, as well as structural features and cross-species conservation. The deep structure of the network (see Figure 3.4) allows it to learn interactions between features as well as model non-linear relationships between the input features and output. More complex neural networks have since been applied to other biological problems, including the prediction of enhancers (Liu et al., 2016; Jia and He, 2016; Kim et al., 2016) or the effect of changes in enhancer regions (Zhou and Troyanskaya, 2015).

An additional benefit of neural networks is their modularity. I mentioned in the section's overview that convolutional neural networks (CNNs) as well as recurrent neural networks (RNNs) are particularly useful for learning from DNA sequences. CNNs and RNNs can also be used together to create a network that combines the benefits of each type of network. One example of this approach being applied to a biological problem is a very recent paper by Angermueller et al. (2017) which predicted per-CpG DNA methylation levels in single cells using a CNN to learn DNA motifs, and an RNN to learn dependencies between methylation levels of CpGs in the same genomic neighbourhood.

3.2.4 Hidden Markov models

A hidden Markov model (HMM) is an unsupervised learning method that can be used for segmenting sequential data. This segmentation assigns a label to each segment, and the process can be thought of as a clustering of the positions in the input sequence into a fixed number of clusters. The model assumes that a sequence of observations $x = (x_1, \dots, x_N)$ can be explained by the hidden state of the system $z = (z_1, \dots, z_N)$, and that this sequence of hidden states is a Markov process where

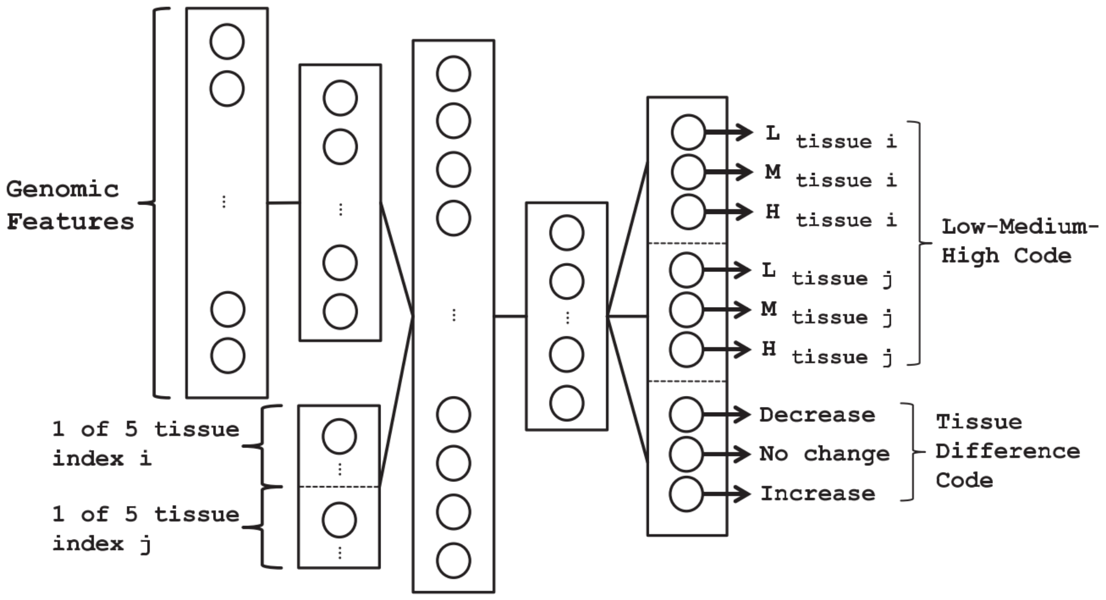


Figure 3.4: Deep network for alternative splicing prediction. The network was created by Leung et al. (2014) to predict the tissue-specific alternative splicing of mRNA transcripts. The output of the network is the approximate percent spliced in (low = 0 - 0.33, medium = 0.33-0.66, high = 0.66-1.0) for each exon and tissue, and an indication of the change in inclusion between two tissues.

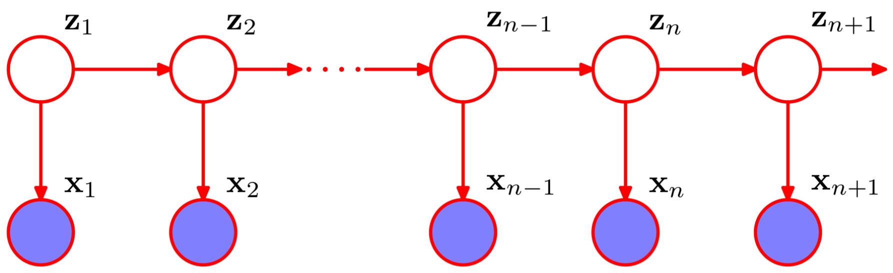


Figure 3.5: Hidden Markov model. Blue nodes signify observations, and the white nodes are hidden states. Arrows indicate conditional dependence relationships between nodes. From Bishop (2006).

the state of the system at any point depends on a fixed number of previous states in the sequence (see Figure 3.5). This means that unlike the other methods we have discussed in this chapter, an HMM does not assume independence between adjacent observations, and we refer to an HMM where the number of previous states M that the model is influenced by as an M-th order hidden Markov model.

The model is defined by five variables: S the set of hidden states, O the set of output symbols, π the initial state probabilities ($\pi = (\pi_1, \ldots, \pi_{|S|})$), B the probability of emitting each output symbol when in a given state ($B_{ik} = p(x_i|z_k)$), A the probability of transitioning between hidden states which is independent of the position in the sequence n ($A_{jk} = p(z_{nk} = 1|z_{n-1,j} = 1)$), where z_{nk} is an indicator variable that is 1 of the hidden state is k at the nth position in the sequence and 0 otherwise). The number of states $|S|$ is usually the only parameter given to the algorithm. Given this set of variables, and assuming we are using a 1st-order HMM, it is possible to determine the probability distribution of hidden states at a given position n in the sequence:

$$p(z_n|z_{n-1}, A) = \prod_{k=1}^{|S|} \prod_{j=1}^{|S|} A_{jk}^{z_{n-1} z_{nk}} \tag{3.22}$$

This is slightly different for the first state z_1, because it does not have a parent node. Instead, we use the initial state probabilities:

$$p(z_1|\pi) = \prod_{k=1}^{|S|} \pi_k^{z_{1k}} \tag{3.23}$$

Also, the probability of a given observation can be calculated as:

$$p(x_n|z_n, B) = \prod_{k=1}^{|S|} p(x_n|B_k)^{z_{nk}} \tag{3.24}$$

These probabilities can then be used to calculate the joint probability distribution over both hidden states Z and observed data X:

$$p(X, Z|\theta) = p(z_1|\pi) \left[\prod_{n=2}^{N} p(z_n|z_{n-1}, A) \right] \prod_{m=1}^{N} p(x_m|z_m, B) \tag{3.25}$$

where $\theta = \pi, A, B$, $X = x_1, \ldots, x_N$, and $Z = z_1, \ldots, z_N$.

We can use this probability distribution to find the most probable hidden state sequence for a set of observations, which is called the Viterbi path:

$$z^{\text{Viterbi}} = \arg\max_z p(Z = z | X = x) \tag{3.26}$$

A similar but different problem is determining the most probable hidden state at a certain position in the sequence z_i. This is called the posterior decoding:

$$z_i^{\text{Posterior decoding}} = \arg\max_{j \in 1, \ldots, k} p(Z_i = j | X = x) \tag{3.27}$$

The parameters for the HMM can be estimated using an expectation maximization algorithm called the Baum-Welch algorithm, but we will not go into detail on that. A description of the algorithm can be found in Bishop, 2006.

Feature importance The type of HMMs we have discussed so far are not usually used with the intent of calculating feature importance. Nevertheless, it is possible to get some information about what variables are most prevalent in each state by looking at the table of emission probabilities (the variable B above, see an example in Figure 3.6).

Biological applications HMMs have been used extensively in the field of computational biology. A non-exhaustive list of biological problems they have been applied to include gene finding (Lukashin, 1998), pairwise and multiple sequence alignment (Eddy, 1995; Durbin et al., 1998), the identification of copy number variants (Love et al., 2011), protein secondary structure prediction (Asai, Hayamizu, and Handa, 1993) and the identification of protein-RNA interactions (Krakau, Richard, and Marsico, 2017).

Particularly relevant to this thesis are the applications of HMMs to CpG islands and general genome segmentation. The textbook application of HMMs to biological data is an HMM for identifying CpG-rich regions using DNA sequence by Durbin et al. (1998). This was an extension of the original HMM for CpG islands by Churchill (1992). The work was later expanded upon by Wu et al. (2010). Another well-known application is ChromHMM, a tool that uses an HMM

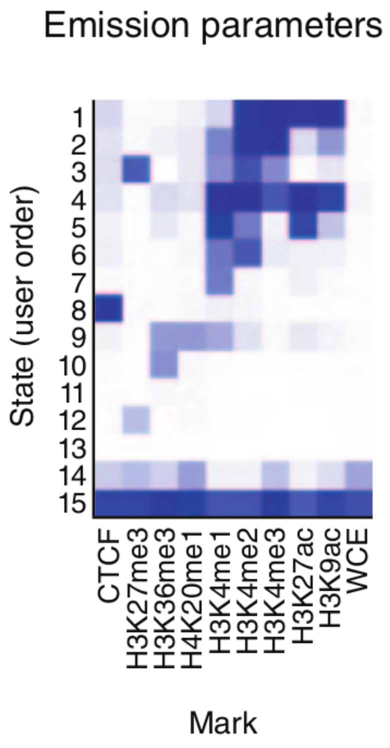


Figure 3.6: Visualization of emission probabilities from ChromHMM. Hidden states are rows and observed ChIP-seq regions are columns. Darker blue indicates a higher emission probability. From Ernst and Kellis (2012).

to segment the genome into different "chromatin states" based on ChIP-seq data (Ernst and Kellis, 2012).

Chapter 4

Predicting non-methylated islands from DNA sequence

Genome-wide methylation of CpG dinucleotides is a hallmark of vertebrate genomes. An exception to this ubiquitous methylation is the set of regions known as non-methylated islands (NMIs), which are stretches of unmethylated genomic DNA. Our understanding of vertebrate NMIs has until recently been limited by the lack of experimental methods which can identify these regions genome-wide. Fortunately, recent advances in high-throughput sequencing as well as experimental methodologies have finally given us the chance to observe NMIs genome-wide in a collection of vertebrates, both warm- and cold-blooded as well as several tissues. In the following chapter, we present the results of a computational analysis of NMIs, including a method for more accurate prediction of NMIs using their DNA sequence. We then use this method to further our understanding of the difference between cold-blooded and warm-blooded NMIs. Additionally, we apply the same method to NMIs in several tissues, both to compare the accuracy of NMI prediction in different tissues and also to attempt to discriminate between tissue-specific NMIs. The content of the chapter is an extended and adapted version of a paper which was published in PLoS Computational Biology (Huska and Vingron, 2016), except for Section 4.4 which was written after the paper was published.

4.1 Motivation

DNA methylation is known to play an important role in vertebrate gene regulation (Bird and Wolffe, 1999; Deaton and Bird, 2011). Most of the human genome is usually methylated, however over 30 years ago a relatively small number of non-methylated regions were identified using methylation-sensitive restriction enzymes (Cooper, Taggart, and Bird, 1983). These non-methylated regions were found to have a higher than expected number of CpG dinucleotides when compared to the rest of the genome, and it was suggested that this is because methylated CpGs are more likely to be mutated to CpAs or TpGs than non-methylated CpGs, leading to the reduction of CpG dinucleotides in most of the genome (Coulondre et al., 1978; Bird, 1980). Due to the increasing availability of genomic sequence data but the lack of a method for the genome-wide measurement of methylation, this sequence-based proxy for non-methylated regions became popular, and they have come to be referred to as CpG islands. One of the first sequence-based definitions of CpG islands was proposed by Gardiner-Garden and Frommer (1987), which defines CpG islands as regions of the genome that have a length of >200 bp, GC content >50%, and a ratio of observed CpGs to expected CpGs (the "CpG Ratio") >0.6. A variant of this method is still used to provide an annotation of CpG islands in the popular UCSC Genome Browser (Kent et al., 2002), and for many years these CpG islands continued to be used as a proxy for non-methylated regions of the genome.

CpG ratios are also used in vertebrates to classify genes into those with high CpG promoters and low CpG promoters, due to the observation that there is a clear bimodal pattern in the CpG ratios of human promoters (Davuluri, Grosse, and Zhang, 2001; Saxonov, Berg, and Brutlag, 2006). This bimodality is also observed in several other vertebrates including humans, chicken, frog and zebrafish (Elango and Yi, 2008). However, the overall percentage of promoters that overlap a CpG island was later shown to be much lower in cold-blooded vertebrates (<20%) when compared to warm-blooded vertebrates (>40%) (Sharif et al., 2010). This suggested that very few promoters are unmethylated in cold-blooded vertebrates, or that the role of CpG-rich regions may differ between cold- and warm-blooded vertebrates.

Later, tissue-specific methylated regions were identified and found to be correlated with tissue-specific gene expression. A study by Song et al. (2005) showed that out of 150 differentially methylated regions that were studied, 100 of the regions overlapped with predicted CpG islands, many of which were unexpectedly shown to be methylated in most tissues. More recent work has reported similar results on a genome-wide scale (Ziller et al., 2013; Mendizabal and Yi, 2015). This dynamic methylation, even at CpG islands, made it clear that some disagreement between true unmethylated regions and CpG island predictions should be expected since CpG island predictions are not tissue-specific. Additionally, differences in GC content of non-methylated regions between different organisms were observed. For example, in contrast to other vertebrates, fish CpG islands were found to not be GC-rich, despite the regions still having a high CpG ratio (Cross et al., 1991). Also, a study of CpG islands in mouse and human showed that the GC content of islands in the two organisms differs (Antequera and Bird, 1993). These findings suggested that a single model of CpG islands for all of these species would not be appropriate, and at the very least the cutoffs for calling a region a CpG island would need to be adapted when making predictions in different organisms.

Fortunately, experimental methods have now been developed that are able to identify methylated or non-methylated regions (also called non-methylated islands, or NMIs) in a given cell line or tissue genome-wide. These methods include whole-genome bisulfite sequencing (Lister et al., 2008; Cokus et al., 2008) as well as the Bio-CAP method (Blackledge et al., 2012), the latter of which is specifically for identifying regions that are unmethylated (see Section 2.3 for more details). Whole genome bisulfite sequencing data is publicly available, including experiments covering ten different human tissues (Mendizabal and Yi, 2015). Additionally, Bio-CAP was recently used to determine the location of NMIs in several vertebrates, including both warm-blooded (human, mouse, chicken and platypus) and cold-blooded (lizard, frog and zebrafish) vertebrates, in multiple tissues including testes and liver (Long et al., 2013). In that study it was noted that the number of non-methylated regions that overlap with predicted CpG islands was in most cases quite low (e.g. $\approx 20\%$ overlap in zebrafish, see Figure 4.1), leading the authors of that study to conclude that the computational methods that are commonly used to identify CpG islands are not able to accurately identify NMIs in vertebrate genomes. The same study (Long et al., 2013) also showed that despite the low percentage suggested by CpG island predictions, over 50% of promoters in all vertebrates that were studied actually do have non-methylated regions at their transcription start sites, but that existing CpG island prediction methods are simply not able to identify them in cold-blooded vertebrates.

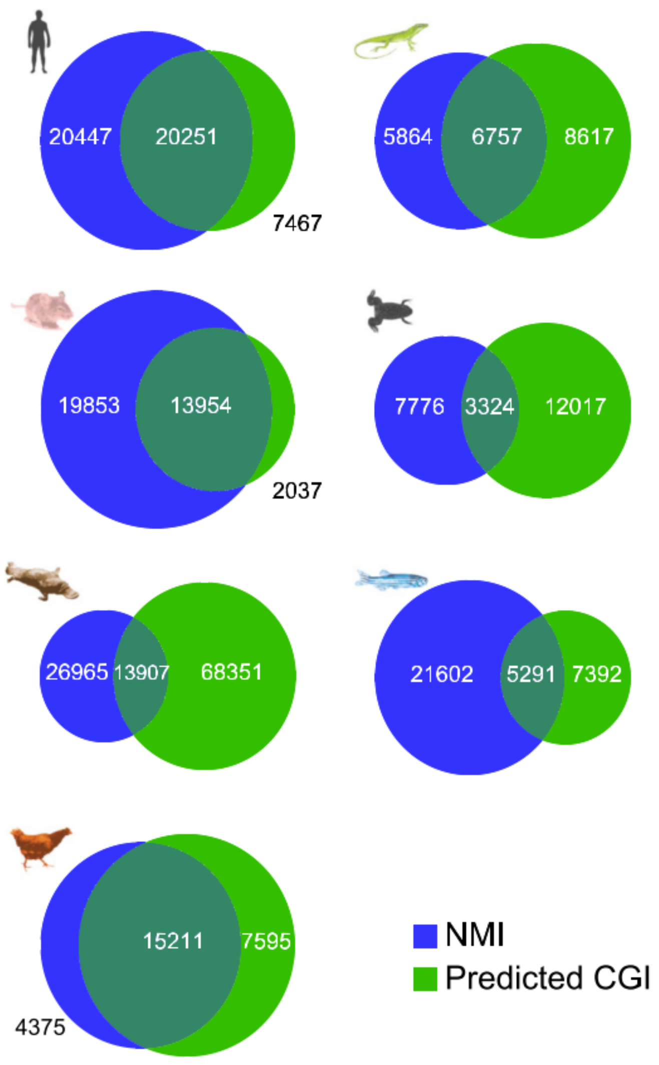


Figure 4.1: Overlap between Bio-CAP NMIs and UCSC CpG islands. The "NMIs" are experimentally determined using the Bio-CAP method and the "Predicted CGIs" are from UCSC's genome browser and are predicted using a method similar to that of Gardiner-Garden and Frommer (1987). Figure from Long et al. (2013).

4.1.1 Computational methods for NMI prediction

We will consider two general approaches to predicting NMI regions from DNA sequence using machine learning: unsupervised or supervised learning approaches (see Section 3.1.2). The choice between these two methods is primarily made based on how many known NMI and non-NMI regions, which we will collectively call labeled data, are available.

As stated earlier, CpG islands have long been used as a proxy for non-methylated islands because of the shortage of experimental methods to identify NMIs, and for the same reason the prediction of CpG islands from DNA sequence is a problem that was typically solved using unsupervised learning methods. These methods cluster the input DNA sequences into two or more groups, and one of these groups is assigned to be the CpG islands after the groups have been assigned, usually based on the CpG content of the sequences in that group. The initial ad-hoc sequence-based definition of Gardiner-Garden and Frommer (1987) (outlined above) includes somewhat arbitrarily chosen cutoffs which can have a considerable effect on which regions are called a CpG island. This motivated the use of a method with a strong statistical basis by Churchill (1992), an unsupervised learning method called a *hidden Markov model*, or HMM (see Section 3.2.4 for a description of HMMs). Churchill's HMM uses each DNA nucleotide in the genome as an observation, with the C position of each CpG dinucleotide encoded as a binary 1, and the rest of the sequence as a 0. This sequence is modeled as a first-order Markov chain, with two hidden states: "normal" and "CpG rich". Later, a similar HMM was outlined in Durbin et al. (1998), but omitted the binary encoding of the sequence and modeled each of the four possible nucleotides separately, using eight hidden states instead of two: a "normal" and "CpG rich" state for each of the four nucleotides (see Figure 4.2). These methods have not been widely used for annotation of the genome for two reasons: first, these models would need to have many more than two states to properly model local variations in base composition, and second, that modeling each nucleotide separately leads to a model that is overly complex when working on the scale of whole genomes (Wu et al., 2010).

More recently, Wu et al. (2010) created an HMM to address these problems. It uses non-overlapping 16 bp genomic windows as observations rather than individ-

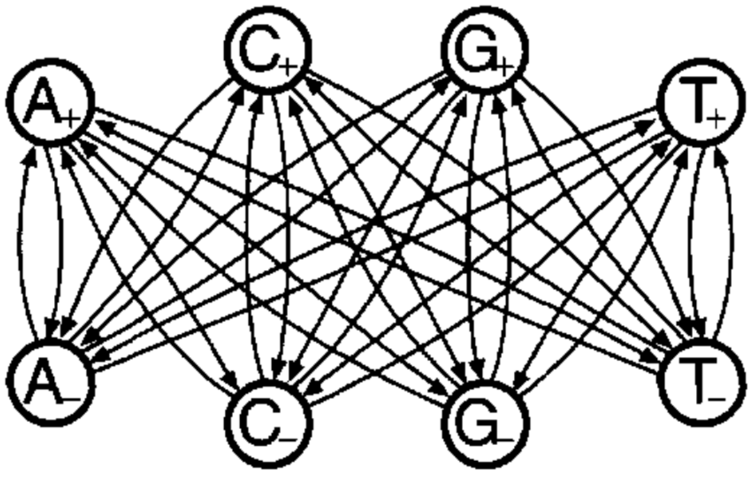


Figure 4.2: Durbin's HMM model for CpG islands. CpG-rich states for each nucleotide are indicated with a + subscript, and normal with a - subscript. Transitions within the same state (CpG-rich or normal) were omitted for clarity. From Durbin et al. (1998).

ual nucleotides, and also takes into account local changes in GC content and its impact on the CpG ratio. These changes led to an increase in the overlap between experimentally determined differentially methylated regions and their CpG island predictions, compared to UCSC CpG islands.

In this study we chose another approach, to predict NMIs using supervised learning. We decided to use this approach because it allows us to directly identify predictive sequence patterns in the newly available experimentally determined non-methylated island regions which we use as training data. This also differs from the previous methods because we do not make any assumptions about the sequence content of the non-methylated regions (for example, the importance of CpG dinucleotides and GC content), but rather we can learn any characteristic sequence features directly from the example NMIs themselves. As a supervised learning method we considered several of the methods outlined in Section 3.2. Deep learning was quickly removed from consideration because at the time the project was started there were no suitable implementations available. Choosing between linear models (specifically logistic regression) and support vector machines, we selected support vector machines because they would allow us to more easily evaluate different string kernels (see Section 3.2.2). The use of different string kernels could potentially be helpful for modeling the binding of transcription factors, which are known to be affected by DNA methylation (Yin et al., 2017) and could also have an effect on methylation. For example, certain string kernels can model mismatches and gaps in the DNA subsequences we are comparing (the mismatch kernel), and also model the binding of proteins as dimers (the gappy pair kernel, see Section 3.2.2 for details).

4.2 Methods

4.2.1 Data

NMIs in six vertebrates. Experimentally determined NMIs from the testes and liver of six vertebrates were taken from Bio-CAP experiments (see Section 2.3.3 for details) performed by Long et al. (2013), and are publicly available from GEO under the accession number GSE43512. The six vertebrates that were analyzed, along with their UCSC genome version, were *Homo sapiens* (hg19), *Mus musculus* (mm9), *Gallus gallus* (galGal3), *Anolis carolinensis* (anoCar2), *Xenopus tropicalis* (xenTro3), and *Danio rerio* (danRer7). Reference genomes were downloaded from UCSC to match these genome versions. The platypus data was omitted because its genome assembly is highly fragmented, consisting of over 200 thousand separate sequence fragments, and only approximately 17% of the organism's genome could be mapped to a chromosome and ordered (Lewin et al., 2009).

Section 4.3 presents the analysis of testes NMIs across these six vertebrate species. The same analysis was repeated in liver and the results were qualitatively the same, so those results have been omitted for the sake of conciseness.

NMIs in 10 human tissues. In order to be able to compare NMIs across a wider range of tissues, NMIs identified using whole genome bisulfite sequencing experiments (see Section 2.3.2 for details) in 10 human tissues were obtained from Mendizabal and Yi (2015). The authors post-processed the WGBS data to identify regions that were non-methylated by selecting regions in the genome that have a moderate number of CpG dinucleotides (at least 10 within each 200 bp window of DNA) and a very low average amount of methylation (average β values < 0.2 within each window). The human genome version that these NMIs are mapped to is hg38. When comparing these NMIs with those from the Bio-CAP NMIs, the coordinates for the Bio-CAP NMIs were converted from hg19 to hg38 coordinates using the `liftOver` tool from UCSC.

4.2.2 Existing CpG island predictions

CpG island predictions from UCSC's Genome Browser were downloaded from the UCSC FTP site. These predictions are based on a variant of the original Garden-Gardiner and Frommer method (Gardiner-Garden and Frommer, 1987), with a number of modifications: the length of the region must be >200 bp, the percent of G or C nucleotides must be >50%, observed/expected CpG ratio must be >0.6, and an additional running score is calculated that must remain above 0 for the entire length of the island. The running score is computed by adding 17 for each CpG, and subtracting 1 for every other base. Also, islands are cut in half at their maximum running score and each half is evaluated separately (for details see `http://genomewiki.ucsc.edu/index.php/CpG_Islands`).

Additionally, CpG island predictions from Wu et al. (2010) were downloaded from the paper's website if available (for *Gallus gallus*, *Mus musculus* and *Homo sapiens*), or calculated from scratch using default parameters (for *Danio rerio* and *Anolis carolinensis*). The one exception is *Xenpous tropicalis*, whose genome repeatedly caused the software to crash and was therefore excluded from the analysis.

Lastly, the CpG observed-expected ratio, or CpG Ratio, is used in several comparisons, because it is the basis for most CpG island predictions. It is defined as:

$$\text{CpG Ratio} = \frac{\text{frequency of CpG's in sequence}}{\text{frequency of C's in sequence} \times \text{frequency of G's in sequence}}$$

4.2.3 Genome preprocessing

For the genome-wide analysis, each genome was pre-filtered to remove regions that are not uniquely mappable using the GEM mappability program (version

20130406-045632, Derrien et al., 2012). This was done because the experimental methods that we used to define our gold standard NMIs are both sequencing-based, and therefore is blind to NMIs that are in regions that are not uniquely mappable. GEM was run with default parameters with the following exceptions, which were chosen in order to match the sequencing data and read mapping settings that were used when defining NMI peak regions from the Bio-CAP experiments (Long et al., 2013): generate an index that includes the reverse complement DNA sequence, use a read length of 51 bp, and allow a maximum of 2 mismatches.

Additionally, a region of 500 bp centered at the borders of each Bio-CAP NMI was removed from the genome-wide analysis. This was done because Bio-CAP is not a high resolution method, and therefore there is some uncertainty about the exact location of the NMI borders (see Section 4.4.4). The border regions were removed so that we could try to avoid training the classifier on mislabeled regions, and to hopefully reduce the number of false positive and false negative regions in the Bio-CAP training sets overall.

4.2.4 Support vector machine implementation

For the genome-wide NMI predictions, we used the very fast implementation of the spectrum kernel SVM that is part of the Shogun machine learning toolbox (Sonnenburg et al., 2010).

For the smaller dataset of 10 human tissue where we only looked at tissue-specific NMIs, we additionally used more computationally expensive kernels: the mismatch and gappy pair kernels (see Section 3.2.2 for details). The implementation that was used in this case was from the R package Kebabs (Palme, Hochreiter, and Bodenhofer, 2015).

4.2.5 Identifying transcription factor motifs

Two methods were used to attempt to identify sequence specific transcription factor motifs that were either enriched or depleted in the sequences that were analyzed. First, the sequences of NMI and non-NMI regions were analyzed using the tool MEME-ChIP (Machanick and Bailey, 2011). This tool performs *ab initio* motif discovery, motif enrichment analysis, motif visualization, binding affinity analysis and motif identification. The motif discovery is performed using the classic MEME algorithm (Bailey et al., 2006), as well as DREME (Bailey, 2011), an algorithm that specifically takes into account a background set (our non-NMI regions).

The second method for identifying transcription factor motifs was using the k-mer weights that can be calculated using a trained SVM classifier, as outlined in Section 3.2.2. These k-mer weights can be thought of as an indication of the importance of those k-mers for predicting the class of a given input DNA sequence. To compare these k-mers to known sequence specific transcription factor binding motifs, the tool Tomtom (Gupta et al., 2007) was used. As a database of motifs to compare against, 519 motifs from the JASPAR 2016 core vertebrates database were used (Sandelin et al., 2004).

4.2.6 Parameter tuning and performance evaluation

Genome-wide predictions Parameter tuning, model training and performance evaluation was carried out separately on each organism and tissue as explained below. Two exceptions to this was in the calculation of the top 20 k-mers for each organism, and the cross-species predictions, in which case the k-mer length parameter was fixed at 6 and SVM soft margin penalty parameter C fixed to 1.0 to make the results more easily comparable across organisms.

In order to select the appropriate parameters for the SVM and to evaluate its performance when identifying NMIs, we split each organism's preprocessed genome into three sets: a parameter tuning set which was used to select optimal model parameters, a separate training set and a test set to evaluate the performance of the model on held out data. A random subset of each organism's chromosomes was set aside as the test set, such that approximately half of the organism's genomic sequence was contained in the test set. Entire chromosomes were set aside rather than individual genomic windows because adjacent windows are not independent, and having adjacent windows in the training and test set could bias the results. The remaining chromosomes were then divided into NMI and non-NMI regions, and those regions were split into 750 bp windows of which a random 50% were used for parameter tuning and the other 50% were used for training. This window size was selected after considering the length distribution of the Bio-CAP NMIs to ensure that the majority of NMIs would be longer than or equal to the window length. In order to control for the differing length of each organism's genome as well as the different ratio of background-to-NMI sequences, the tuning and training windows were subset to have a total of 30,000 windows, with a 5:1 ratio between background and NMI sequences.

In order to identify the optimal parameters of the SVM model we chose to use a common and simple method called a *grid search*, which involves evaluating the SVM's performance on a subset of the data for every possible combination of parameters using cross-validation. In this case, the SVM's soft margin penalty C as well as the k-mer length were selected in this way. A grid search was performed over C values of 0.01, 0.1 and 1, and k-mer lengths between 2 and 9. 5-fold cross-validation was used on the parameter tuning dataset and the parameters that yielded the highest average AUROC were selected (the best parameters are shown in Table 4.3). The R package caret (Kuhn, 2008) was used for parameter tuning.

The SVM was then trained on the training set using the selected parameters, and the chromosomes in the test set were split into 750 bp windows whose NMI status was then predicted. For comparison with other methods, the windows were further split into 50 bp windows, because the predictions of the other methods are occasionally quite short (e.g. the Wu HMM method predictions have a small percentage of predictions less than 100 bp), and these predictions would be lost if the window size is too large. A window was called a true NMI if $\geq 50\%$ of its length was covered by an experimentally determined NMI.

This entire process was carried out 5 times with different random seeds, to estimate how much variance in the selected parameters and overall predictive performance we see on different random subsets of chromosomes, different random splits into training and test sets, and different subsampled sets.

When performing cross-species prediction, we reused the same training and test sets that were defined for the intra-species prediction. This ensures that the performance of cross-species prediction can be more directly compared to the intra-species prediction because they both use the exact same training and test sets, and therefore preserve several properties that could affect training performance including training set size, any possible chromosomal biases, and training and test set class imbalance.

Direct tissue-specific NMI comparison When comparing NMIs from different tissues directly, we used a simpler 10-fold cross-validation approach rather than splitting the genome up by chromosome. This is because we do not have to worry as much about the dependence between adjacent windows because we did not split the NMIs into windows in this case, and we also do not include non-NMI genomic regions. Parameter tuning using a grid search and 5-fold cross-validation was performed for each one of the 10 cross-validation iterations separately.

4.2.7 Randomly trained model

When performing genome-wide predictions, in order to control for overfitting, as stated above we used non-overlapping parameter tuning, training and test datasets. Optimal parameters were then chosen using 5-fold cross-validation on the parameter tuning set. In addition, we also wanted to evaluate how the model behaves when trained on data with randomly shuffled labels. This extra check was carried out to ensure that we are not leaking information between any of the three previously mentioned sets of data, and also to evaluate the behaviour of the model when trained on a dataset with a different level of class imbalance than the final test set (5:1 in the parameter tuning and training sets versus approximately 20:1 in the test set). By using randomly shuffled class labels in the training set we are able to preserve the relative class abundance and exact number of training examples in each class.

We first looked at the results of parameter tuning when training on shuffled labels. We saw that the AUROC (the criteria which is being used to choose the best parameters) when performing 5-fold cross-validation on the parameter tuning set is very close to 0.50 regardless of which parameters are selected (data not shown). It is noteworthy to mention that in this setting the class imbalance is the same in both the training and test set, because this training and test set is being drawn from the 5:1 downsampled parameter tuning set.

We then fixed the parameters to match those that were used for cross-species prediction (k-mer length $= 6$, $C = 1.0$), trained on the large held out training dataset with shuffled labels and predicted on the held out test set. Unlike the parameter tuning setting described above, the class imbalance is not the same in the training and test sets. The training set has a 5:1 ratio of background:NMI sequences, because the background sequences were downsampled to keep computational runtime reasonable. In contrast, the test set has an imbalance that differs from one organism to the next, since the test set is made up of approximately half of the organism's genome without any subsampling, and therefore has the same ratio of background:NMI sequences as the organism's genome does. The results show that the test set AUROC is approximately 0.35-0.65 for this classifier, likely reflecting the effect of a different amount of class imbalance between the training and test set. AUPRCs are close to the value of approximately 0.015 that we would expect from a classifier which randomly assigns a class. These results are shown in Figure 4.3 and Figure 4.4 (ROC and PRC plots respectively), as well as Table 4.1 as "SVM (random)".

4.2.8 Software availability

A set of scripts were produced that can be used to train a classifier and predict NMIs in a set of DNA sequences. This code is available at `https://github.com/matthuska/predict-nmi`, and is licensed under the MIT open source license. These scripts use the spectrum kernel SVM from the Shogun machine learning toolbox (Sonnenburg et al., 2010).

4.3 Identification of NMIs genome-wide in six vertebrates

In this section we take advantage of newly available genome-wide experimentally determined NMIs from Long et al. (2013) to answer several questions. First, how well can we predict experimentally determined NMIs genome-wide using DNA sequence, and what are the important sequence features for this prediction? We also compare these results between all six vertebrates. Finally, we look at how well we can predict NMIs when we train in one species and predict in another. Unless otherwise stated, the NMIs being used are from Bio-CAP testes experiments.

4.3.1 DNA sequence is highly predictive of non-methylated islands

To investigate how informative DNA sequence is in determining whether or not a given genomic region is a non-methylated island, we used a subset of each organism's chromosomes to train an SVM to differentiate between known NMI sequences from testes and the rest of the genome (see the Methods section for details). This classifier was then used to predict NMIs in the rest of the genome and its performance was evaluated. This process was carried out separately for each organism, which allowed us to identify differences in predictive performance and important sequence features between the six species. As a baseline, the performance of the SVM classifier was compared to the observed versus expected CpG ratio of the sequence, as well as to two existing CpG island predictions: UCSC's predictions based on the Gardiner-Garden and Frommer method (Gardiner-Garden and Frommer, 1987) and a more recent hidden Markov model-based method (Wu et al., 2010). We compare the performance of the methods using ROC curves and Precision-Recall curves in order to understand the performance of the classifiers when classifying a given random genomic window as an NMI or not, as well as how well the classifiers can annotate the entire genome while controlling the number of false positives.

The ROC curves show that the SVM classifier was able to identify NMIs based on sequence alone with high predictive performance (see Figure 4.3, Table 4.1 and Table 4.2). A simple CpG ratio is also quite predictive in humans, mice, and chickens (AUROC 0.96–0.98), though the SVM performs slightly better in all cases (AUROC 0.98–0.99). In cold-blooded vertebrates the SVM shows a more marked improvement in predictive performance over the CpG ratio (SVM AUROC 0.91–0.98 versus CpG ratio AUROC 0.76–0.88). The UCSC and Wu HMM methods are both very conservative with their predictions. While they both maintain low false positive rates, they also have very low average true positive rates of between 7% and 49%. The two methods also achieved higher true positive rates in warm-blooded vertebrates than cold-blooded vertebrates, with an average of 3 times more true positives being identified in warm-blooded vertebrates than in cold-blooded vertebrates.

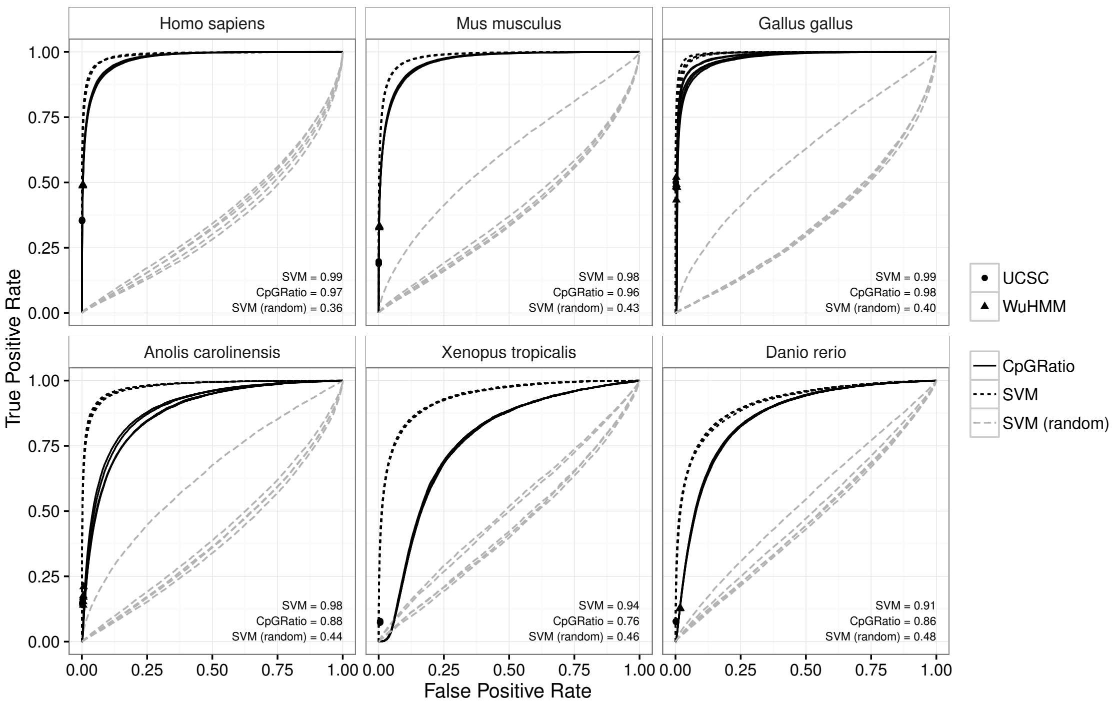


Figure 4.3: ROC curves showing NMI prediction performance. A receiver operating characteristic curve for four different classifiers: SVM (the spectrum kernel SVM), the CpG ratio, UCSC CpG island predictions, the HMM from Wu et al. (2010), as well as an SVM trained on sequences with randomly shuffled labels, "SVM (random)". The UCSC and Wu HMM methods are shown as points rather than curves, because they only provide a set of genomic windows rather than scores for the whole genome, essentially the same as choosing a single cutoff score for the other methods. The prediction was run five times with different random splits of training and test data, therefore five lines or points are shown for each method. The performance is very stable between runs, with the lines for each run almost perfectly overlapping. The average area under the curve across all 5 random splits is indicated in each panel.

Table 4.1: AUROC and AUPRC of NMI prediction genome-wide. Values are the average AUROC and AUPRC of whole genome NMI prediction across 5 random splits of parameter tuning, training and test data. The performance of an SVM trained on sequences with randomly shuffled labels is included as “SVM (random)”.

	AUROC			AUPRC		
organism	CpGRatio	SVM	SVM (random)	CpGRatio	SVM	SVM (random)
Homo sapiens	0.974	0.988	0.356	0.584	0.820	0.013
Mus musculus	0.964	0.983	0.425	0.611	0.774	0.024
Gallus gallus	0.975	0.993	0.397	0.433	0.849	0.019
Anolis carolinensis	0.883	0.977	0.443	0.105	0.664	0.014
Xenopus tropicalis	0.764	0.937	0.462	0.029	0.400	0.012
Danio rerio	0.857	0.913	0.477	0.115	0.416	0.023

Table 4.2: Performance measures for all methods of NMI prediction at a single score cutoff. The Wu HMM and UCSC methods are provided as windows, so it is not possible to compute AUROC and AUPRC curves for them. Therefore, we present several other common performance measures, and apply a score cutoff to the CpGRatio and SVM methods in order to make them comparable. The SVM decision value cutoff was set to 0, and the CpG Ratio score cutoff was 0.6, to match the original cutoff used in (Gardiner-Garden and Frommer, 1987). FPR is the false positive rate and TPR is the true positive rate. The performance of the Wu HMM method for *Xenopus tropicalis* was intentionally left out, because the method consistently crashed when trying to calculate predictions for this genome. All values are averages across 5 random splits of the genome into parameter tuning, training and test sets. Recall is not included because it is identical to the true positive rate.

	FPR				TPR				Precision			
organism	CpGRatio	UCSC	WuHMM	SVM	CpGRatio	UCSC	WuHMM	SVM	CpGRatio	UCSC	WuHMM	SVM
Homo sapiens	0.006	0.001	0.003	0.009	0.557	0.354	0.489	0.810	0.632	0.906	0.713	0.603
Mus musculus	0.001	0.000	0.002	0.016	0.316	0.191	0.329	0.791	0.867	0.982	0.764	0.550
Gallus gallus	0.012	0.002	0.004	0.012	0.734	0.489	0.481	0.864	0.528	0.829	0.719	0.598
Anolis carolinensis	0.032	0.002	0.005	0.016	0.382	0.147	0.170	0.782	0.139	0.512	0.294	0.400
Xenopus tropicalis	0.113	0.006	-	0.024	0.323	0.075	-	0.613	0.036	0.148	-	0.247
Danio rerio	0.270	0.002	0.018	0.002	0.841	0.077	0.128	0.099	0.070	0.536	0.148	0.798

4.3.2 Support vector machine is uniquely able to predict NMIs genome-wide in all six vertebrates

The Precision-Recall curves more clearly show that the SVM outperforms the other methods. Despite the fact that the ROC curves for the SVM and CpG Ratio looked quite similar the PRC shows that the SVM is much better at controlling false positives genome-wide (see Figure 4.4). In the case of all cold-blooded vertebrates (lizard, frog and zebrafish) the CpG ratio-based predictions consist almost exclusively of false positives, regardless of the scoring cutoff that is used. In contrast, the SVM-based method is still able to control the amount of false positives, though at an admittedly low recall.

The UCSC and Wu HMM methods both achieve high precision in warm-blooded vertebrates (>0.71 in all cases), but their precision drops in cold-blooded vertebrates. The UCSC method still manages an average of 0.52 precision in lizard and zebrafish, but this decreases to 0.15 in frog. The Wu HMM method has lower performance, with an average precision of 0.3 in lizard and 0.15 in zebrafish. The Wu HMM software was not able to produce a prediction for frog (the software crashed repeatedly). While some of the false positives which lead to low precision for many of the predictions could in fact be false negatives from the Bio-CAP method, we have no way of knowing if this is the case from the Bio-CAP data alone.

4.3.3 Longer CpG-poor sequence features are required for accurate NMI identification in cold-blooded vertebrates

We next sought to investigate why CpG island prediction methods based on CpG ratios perform so poorly at predicting NMIs in cold-blooded vertebrates in comparison to their higher predictive accuracy in warm-blooded vertebrates. Additionally, we wanted to understand why CpG ratios perform so poorly in cold-blooded vertebrates while the SVM classifier is better able to control false positives. To address these questions we looked into the performance of the SVM on the parameter tuning subset of the data, which contained the same number of windows for all

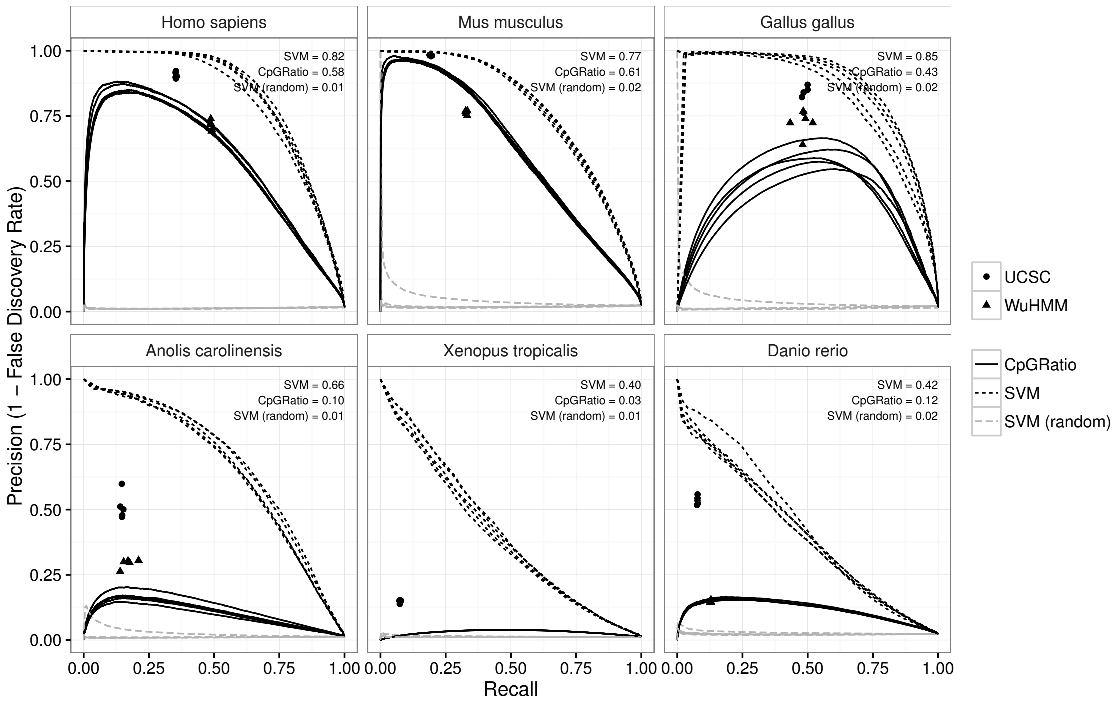


Figure 4.4: Precision-Recall curves of NMI prediction performance. These Precision-Recall curves plot the relationship between the fraction of correctly identified regions (precision) versus the fraction of all NMIs that are identified (recall). The average area under the curve across all 5 random splits of the genome into parameter tuning, training and test sets is indicated in each panel. The performance of an SVM classifier trained on sequences with randomly shuffled labels, "SVM (random)", is shown in grey.

organisms (30,000) and a fixed ratio of non-NMI windows to NMI windows (5:1). For all datasets we calculated the AUROC and AUPRC when using k-mers of increasing length (see Methods section) as features (see Figure 4.5). The results show that short k-mers are already very predictive of NMI status in warm-blooded vertebrates (AUROC >0.97), but that longer k-mers are required for reasonable performance in cold-blooded vertebrates, especially frogs and zebrafish (see Table 4.3). Using longer k-mers that led to a high AUROC in all organisms we observed that the 20 highest scoring k-mers were almost completely devoid of A/T nucleotides in warm-blooded vertebrates, while nearly every k-mer in cold-blooded vertebrates contained an A or T nucleotide (see Table 4.4). Two example regions in frog and lizard are shown in Figure 4.6, where in some windows overlapping an NMI the CpG ratio is low, the UCSC and Wu HMM methods both perform poorly, but the SVM classifier is still able to correctly identify the majority of the NMI region.

The importance of longer, more complex sequence features, especially in frog and zebrafish, suggests that different mechanisms for the establishment and maintenance of non-methylated regions may be dominant in these organisms. To investigate this, the k-mers that contribute the most to the classification of NMIs in each organism were compared to all known transcription factor binding motifs in the JASPAR 2016 vertebrates database (Sandelin et al., 2004) (see Supplementary Table A.1 in Appendix A). In warm-blooded vertebrates as well as lizard, the resulting enriched transcription factor motifs included a number of zinc finger proteins and other DNA binding proteins with GC-rich binding motifs such as SP1, SP2, and E2F family proteins (see some examples in Figure 4.7), as had been previously observed (Deaton and Bird, 2011). In the remaining cold-blooded vertebrates there were essentially no enriched motifs within NMI regions. Direct analysis of the NMI and non-NMI sequences using MEME-ChIP did not identify any enriched motifs that are similar to known TF binding motifs, using the same JASPAR database.

Table 4.3: Optimal SVM parameters for each organism. For 5 separate splits of the genome into parameter tuning, training, and test sets, the optimal parameters (the soft margin penalty C and k-mer length) were determined using 5-fold cross-validation. The parameters were selected based on average AUROC.

Organism	Replicate	AUROC	C	K-mer Length
Homo sapiens	1	0.99	1.00	4
Homo sapiens	2	0.99	0.10	5
Homo sapiens	3	0.99	1.00	4
Homo sapiens	4	0.99	1.00	4
Homo sapiens	5	0.99	1.00	4
Mus musculus	1	0.98	1.00	4
Mus musculus	2	0.98	1.00	4
Mus musculus	3	0.98	1.00	4
Mus musculus	4	0.98	1.00	4
Mus musculus	5	0.98	1.00	4
Gallus gallus	1	0.99	1.00	3
Gallus gallus	2	0.99	0.10	3
Gallus gallus	3	0.99	0.10	3
Gallus gallus	4	0.99	0.10	3
Gallus gallus	5	0.99	1.00	3
Xenopus tropicalis	1	0.93	1.00	7
Xenopus tropicalis	2	0.92	1.00	7
Xenopus tropicalis	3	0.92	1.00	6
Xenopus tropicalis	4	0.92	1.00	6
Xenopus tropicalis	5	0.92	1.00	7
Danio rerio	1	0.91	0.10	7
Danio rerio	2	0.90	1.00	8
Danio rerio	3	0.90	0.10	7
Danio rerio	4	0.90	0.10	7
Danio rerio	5	0.90	0.10	7
Anolis carolinensis	1	0.97	1.00	8
Anolis carolinensis	2	0.97	1.00	8
Anolis carolinensis	3	0.97	1.00	8
Anolis carolinensis	4	0.97	1.00	7
Anolis carolinensis	5	0.97	1.00	8

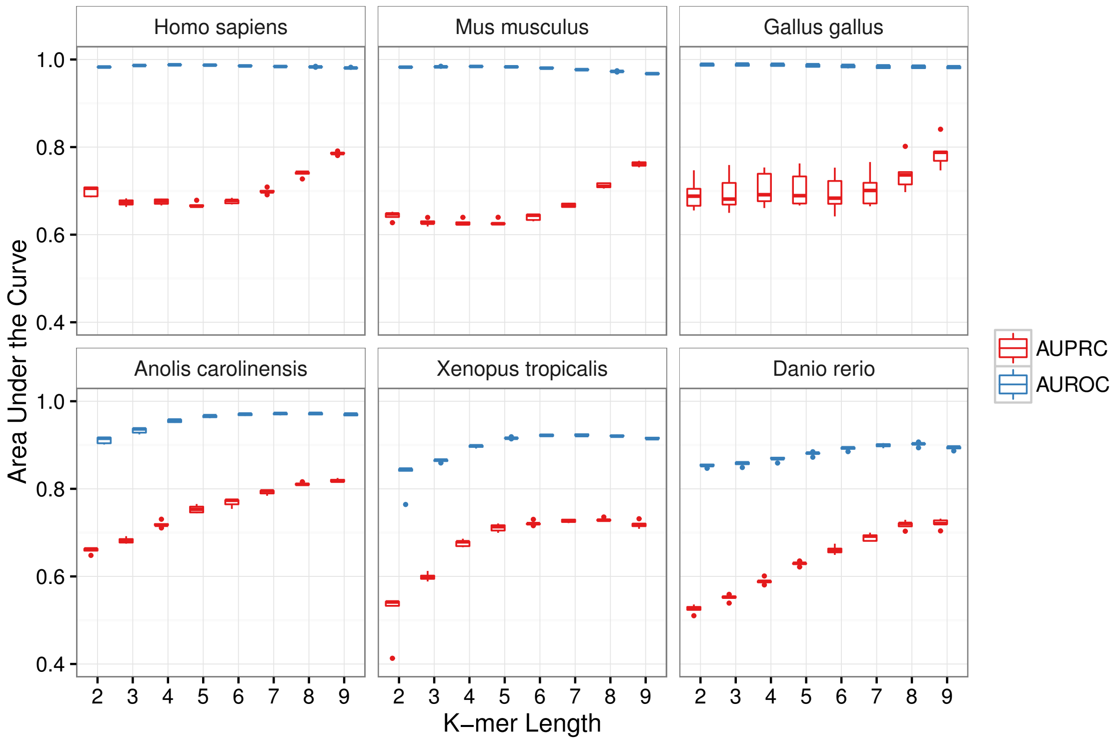
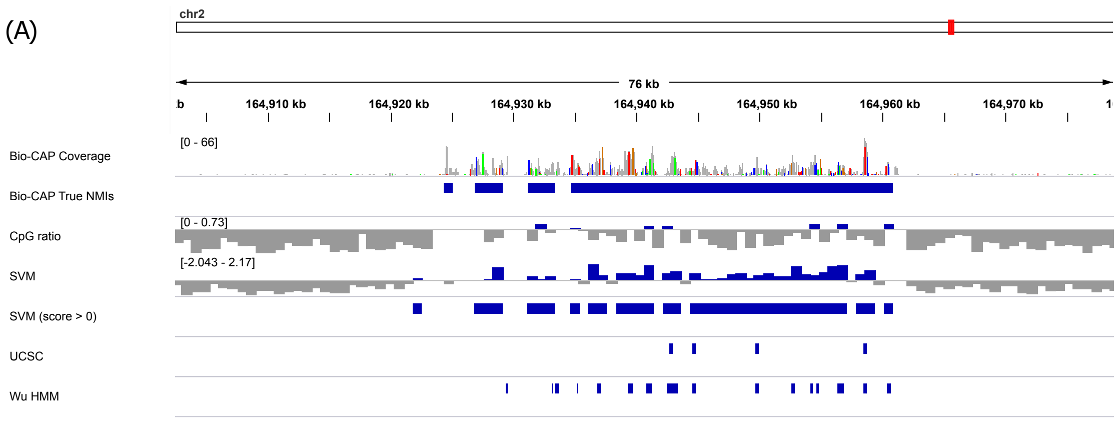


Figure 4.5: AUPRC and AUROC for the SVM with increasing k-mer lengths. While the frequency of di- and tri-nucleotides is already highly predictive of NMI status in warm-blooded vertebrates (AUROC >0.97), the frequencies of k-mers of length 6 or more are required for accurate prediction of NMIs in cold-blooded vertebrates. Box plots show the prediction performance on the parameter tuning set across 5 runs of 5-fold cross-validation. The datasets for all organisms consist of 30,000 750 bp windows with a 5:1 ratio of non-NMI windows to NMI windows. This fixed number of windows and fixed class imbalance means that both the AUROC (blue) and AUPRC (red) can be compared across organisms.

(A)



(B)

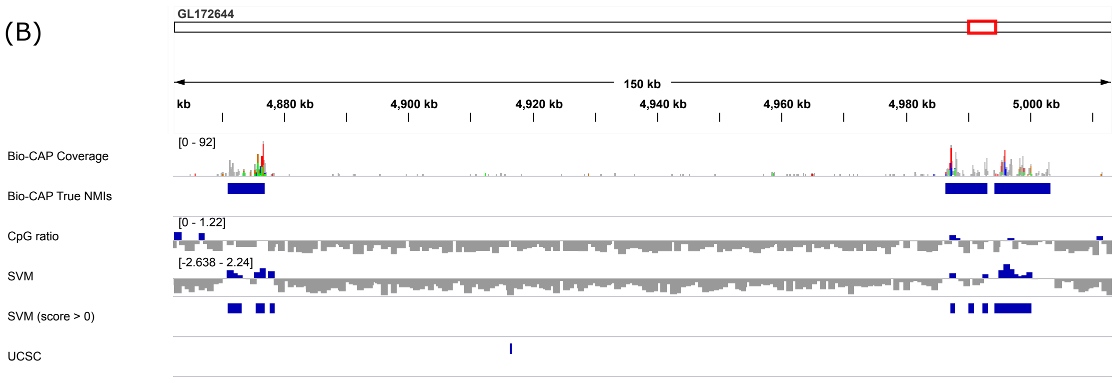


Figure 4.6: Example genomic regions showing prediction improvement with SVM. Two example regions from (A) *Anolis carolinensis* and (B) *Xenopus tropicalis*. In both cases there are NMIs that contain stretches of relatively low CpG content, which are either poorly predicted (in lizard) or not predicted at all (in frog) using CpG ratios, UCSC CpG island predictions or the Wu HMM method. Nevertheless, they are quite accurately predicted using the SVM-based method that uses longer k-mers as features.

Table 4.4: The top 20 highest weight 6-mers. The highest ranked k-mers contribute the most to classifying a region as a non-methylated island. 6-mers containing more than one A or T nucleotide are in bold. A and T nucleotides are almost completely absent from high scoring 6-mers in warm-blooded organisms, while nearly every high scoring 6-mer contains an A or T in cold-blooded organisms.

	Homo sapiens	Mus musculus	Gallus gallus	Anolis carolinensis	Xenopus tropicalis	Danio rerio
1	CCCCGC	GGGCGG	GGGGGG	**TGTGTG**	**ATCTAT**	GCGCGC
2	CCGCCC	CCGCCC	CCCCCC	**AAAAAA**	**ATAGAT**	CGCGCG
3	GCGGGG	CGCCCC	**AAAAAA**	**TCTCTC**	**GATAGA**	**ACACAC**
4	GGGGCG	GGGGCG	**TTTTTT**	**GTGTGT**	**TCTATC**	**GTGTGT**
5	CCCGCC	CCCGCC	GCGGGG	CCCCCC	**ACACAC**	**TAACGT**
6	CGCCCC	GCGCGC	CCCCGC	**AGAGAG**	**GTGTGT**	**ACGTTA**
7	GGCGGG	CCCCGC	GCAGCG	**GAGAGA**	**ATATAT**	**AACGTT**
8	GGGCGG	**AAAAAA**	GCCCCG	**CTCTCT**	**TAGATA**	**TTTAAA**
9	GCGCGC	GCGGGG	CGCTGC	**CACACA**	**TATCTA**	**CACACA**
10	CGCGCC	GGCGGG	CAGCGC	GGGGGG	**CACACA**	**TTAAAT**
11	GGCGCG	CGCGCG	CGCAGC	**TTTTTT**	**TGTGTG**	**TCTCTC**
12	GCGCGG	GCCCCG	CGGGGC	**ACACAC**	**AGATAG**	**CTCTCT**
13	CCGCGG	CTCCGC	GCTGCG	**ATATAT**	**CTATCT**	**TGTGTG**
14	CCGCGC	CCCCCC	GCGCTG	**CCTTCC**	**TATATA**	CGCGCT
15	CGCGGG	CTGCGC	GGCAGC	**GGAAGG**	**AAAAAA**	ACGCGC
16	CCCGGG	CCCGGG	GCTCCG	**GAGGAG**	**TTTTTT**	**GAGAGA**
17	CCCGCG	GCGGAG	CCCGGC	GCGCGC	**AGCAGC**	AGCGCG
18	CTCCCG	CGGGGC	GGGCGG	**CCTCCT**	**GCTGCT**	**TATCTA**
19	GCGGAG	GCTGCG	GCCGGG	**GGAGGA**	**GCAGCA**	**AGAGAG**
20	GCCCCG	GGGCGC	CAGCCC	**AGGAGG**	**TGCTGC**	**ATTTAA**

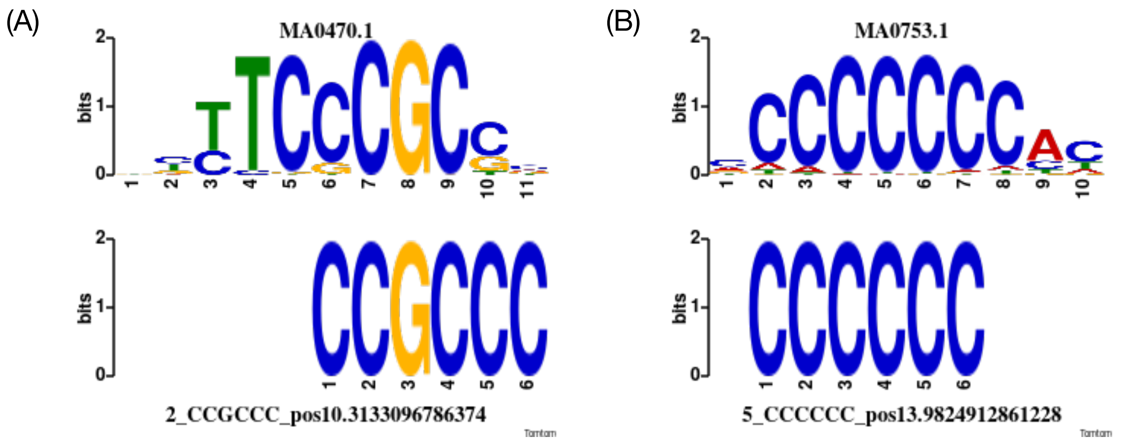


Figure 4.7: Example TF motifs identified using highly weighted *k*-mers. (A) A highly weighted k-mer `CCGCCC` from mouse and human is similar to the TF motif for E2F4. (B) A highly weighted k-mer `CCCCCC` from lizard is similar to the TF motif for SP1.

4.3.4 Cross-species prediction

We also investigated how well a classifier trained in one organism can predict NMIs in another organism. This is particularly useful because it would potentially allow us to improve NMI annotation in species whose genomic sequence is known but genome-wide methylation experiments such as Bio-CAP or whole-genome bisulfite sequencing have not yet been performed, which is the majority of species. In order to make it easier to compare the results across organisms, we fixed the SVM parameters and used a k-mer length of 6 for all organisms. This way we can attribute any differences that we observe to the differences between organisms rather than differences in parameters. Additionally, after observing the clear grouping of samples into cold and warm-blooded vertebrates in the previous results, we decided to add two more training sets to the analysis: a set of sequences sampled from all warm-blooded vertebrate species (human, mouse and chicken), and a set of sequences from all cold-blooded species (lizard, frog and zebrafish).

As shown in Table 4.5, the best predictions (AUPRC) are always achieved when training and testing on the same species. The pooled samples which include the test organism consistently achieve the second best performance. The one exception to this was in chicken, where the prediction based on pooled warm-blooded sequences was slightly better than the predictions using chicken sequences alone, though this difference is very small (0.003 AUPRC). In general the warm-blooded vertebrates are all fairly well predicted by other warm-blooded vertebrates as well as by lizard. A classifier trained on frog and zebrafish sequences performed relatively poorly when trying to predict NMIs in any of the warm-blooded vertebrates, though pooled cold-blooded sequences managed to achieve a high AUPRC in chicken (0.792). Cross-species prediction of NMIs in cold-blooded species was overall very difficult. The only partial exception to this is the lizard, whose NMIs were predicted modestly well (> 0.3 AUPRC) by classifiers trained in any organism or set of organisms, with the exception of frog and zebrafish.

These results also show that the best cross-species predictor for warm-blooded organisms is always the organism that has the most recent common ancestor: mouse and human are more closely related than chicken, and also achieve the highest cross-species predictive performance. Additionally, the predictor based on

Table 4.5: AUPRC of cross-species prediction. Rows are the organisms which the classifier was trained on, and columns are the organisms which the classifier was tested on. The last two rows show the performance when training on a set of combined sequences from human, mouse and chicken (All warm-blooded), or lizard, frog and zebrafish (All cold-blooded). Each value is the mean AUPRC across 5 separate training sets and test sets (see Methods for details). The tissue in all cases was testes.

	H. sapiens	M. musculus	G. gallus	A. carolinensis	X. tropicalis	D. rerio
H. sapiens	0.815	0.709	0.829	0.339	0.047	0.107
M. musculus	0.747	0.752	0.831	0.332	0.052	0.096
G. gallus	0.685	0.635	0.835	0.330	0.088	0.092
A. carolinensis	0.702	0.548	0.770	0.658	0.059	0.088
X. tropicalis	0.392	0.351	0.598	0.254	0.410	0.086
D. rerio	0.497	0.349	0.665	0.164	0.039	0.400
All warm-blooded	0.781	0.726	0.838	0.353	0.059	0.098
All cold-blooded	0.664	0.547	0.792	0.564	0.183	0.217
CpG Ratio	0.584	0.611	0.433	0.105	0.029	0.115

pooled warm-blooded vertebrate species would be useful for predicting NMIs in other warm-blooded vertebrates as well as lizard, a species whose last common ancestor with the three warm-blooded organisms was roughly 300 million years ago. This is not the case in the remaining cold-blooded vertebrates, where there is no trend of better cross-species prediction when training on more evolutionarily close organisms.

4.4 NMIs in different cell types and tissues

Unlike classical CpG islands, which are defined purely based on sequence and therefore do not differ from one cell type to the next, non-methylated regions that are experimentally determined can vary between cell types. This is clearly seen in the dataset we used throughout the previous section from Long et al. (2013), where the authors identified NMIs in both liver and testes in all organisms that were studied. Comparing the locations of NMIs in these two tissues in all organisms, we see that there is a modest portion of NMIs that is not shared in both tissues (see Figure 4.8). This brings up two questions: first, can the DNA sequences of the NMIs in one tissue be used to predict NMIs in another tissue,

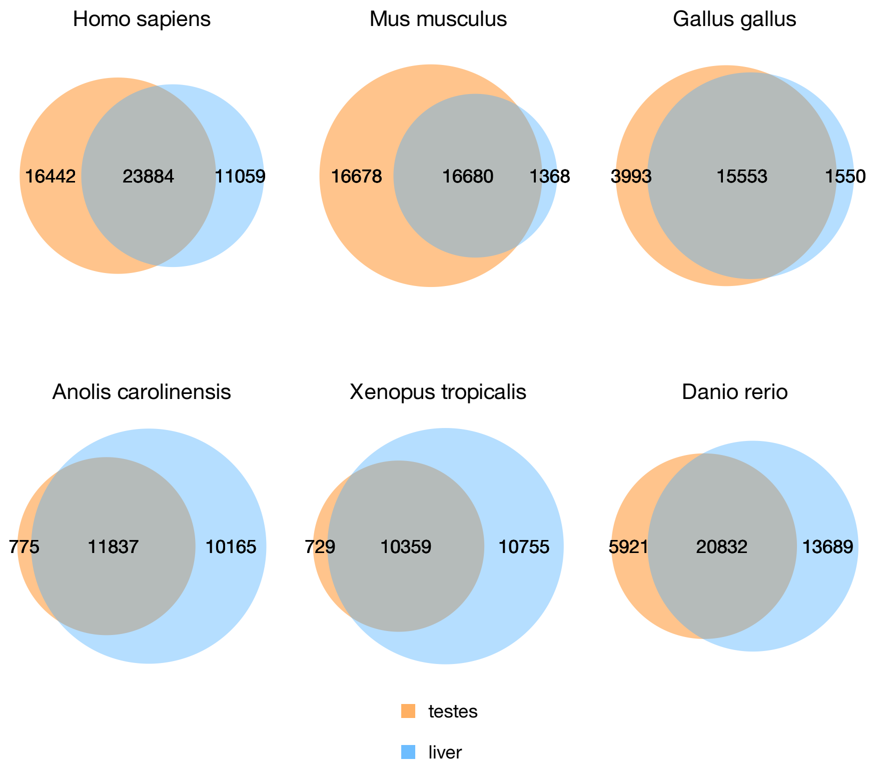


Figure 4.8: Overlap between liver and testes NMIs in six vertebrates. A region is placed in the intersection if it overlaps with a region in the other tissue by at least one base pair.

and second, for NMIs that are tissue-specific can the NMI's sequence be used to determine which tissue or tissues it is active in. In the following text we address these questions in detail.

4.4.1 Cross-tissue prediction in six vertebrates

We know that methylation can be involved in tissue-specific gene regulation. Now that we have experimental data from multiple species that contains non-methylated regions which differ between tissues (see Figure 4.8), it would be interesting to know whether the DNA sequence of NMIs in one tissue can be used to predict those in another tissue in the same species. To answer this question, we used the experimentally determined testes and liver NMIs from Long et al. (2013)

Table 4.6: AUROC of cross-tissue prediction. Scores are the mean AUROC across 5 random splits of the data.

Tissue (train)	Tissue (test)	Hsapiens	Mmusculus	Ggallus	Acarolinensis	Xtropicalis	Drerio
liver	liver	0.986	0.952	0.987	0.968	0.929	0.913
testes	liver	0.980	0.951	0.986	0.959	0.930	0.912
testes	testes	0.986	0.979	0.989	0.975	0.933	0.901
liver	testes	0.981	0.971	0.987	0.973	0.926	0.893

Table 4.7: AUPRC of cross-tissue prediction. Scores are the mean AUPRC across 5 random splits of the data.

Tissue (train)	Tissue (test)	Hsapiens	Mmusculus	Ggallus	Acarolinensis	Xtropicalis	Drerio
liver	liver	0.792	0.668	0.762	0.614	0.371	0.408
testes	liver	0.788	0.662	0.750	0.625	0.409	0.414
testes	testes	0.820	0.755	0.811	0.657	0.406	0.401
liver	testes	0.772	0.724	0.813	0.613	0.354	0.386

as a training set, then trained a classifier on the DNA sequence of the NMIs from each tissue separately, and used that classifier to try to predict NMIs in the other tissue. This was performed for all six species independently so that we can see if there are species-specific differences in cross-tissue predictive performance.

The results show that regardless of species, cross-tissue prediction is nearly as accurate as prediction within the same tissue (see Tables 4.6 and 4.7). The high predictive accuracy of cross-tissue prediction suggests that in all six vertebrates, the DNA sequence features of NMIs in these two tissues are similar. Looking at the importance of all k-mers in the prediction across 5 different random sets of training, test and model selection sets, we see that the correlation of the k-mer importances is relatively low between replicates, but even lower between tissues (see Figure 4.9). We also looked for individual k-mers that were differentially important in the two tissues and found that out 4096 possible 6-mers, across all six vertebrates, just three had a statistically significant difference between the two tissues: `GCTAAG` in zebrafish, and `ATCATC` and `TTTACC` in humans ($n = 5$, paired Student's t-test, corrected for multiple testing using the method of Benjamini and Hochberg (1995). K-mers were considered significant if FDR < 0.1).

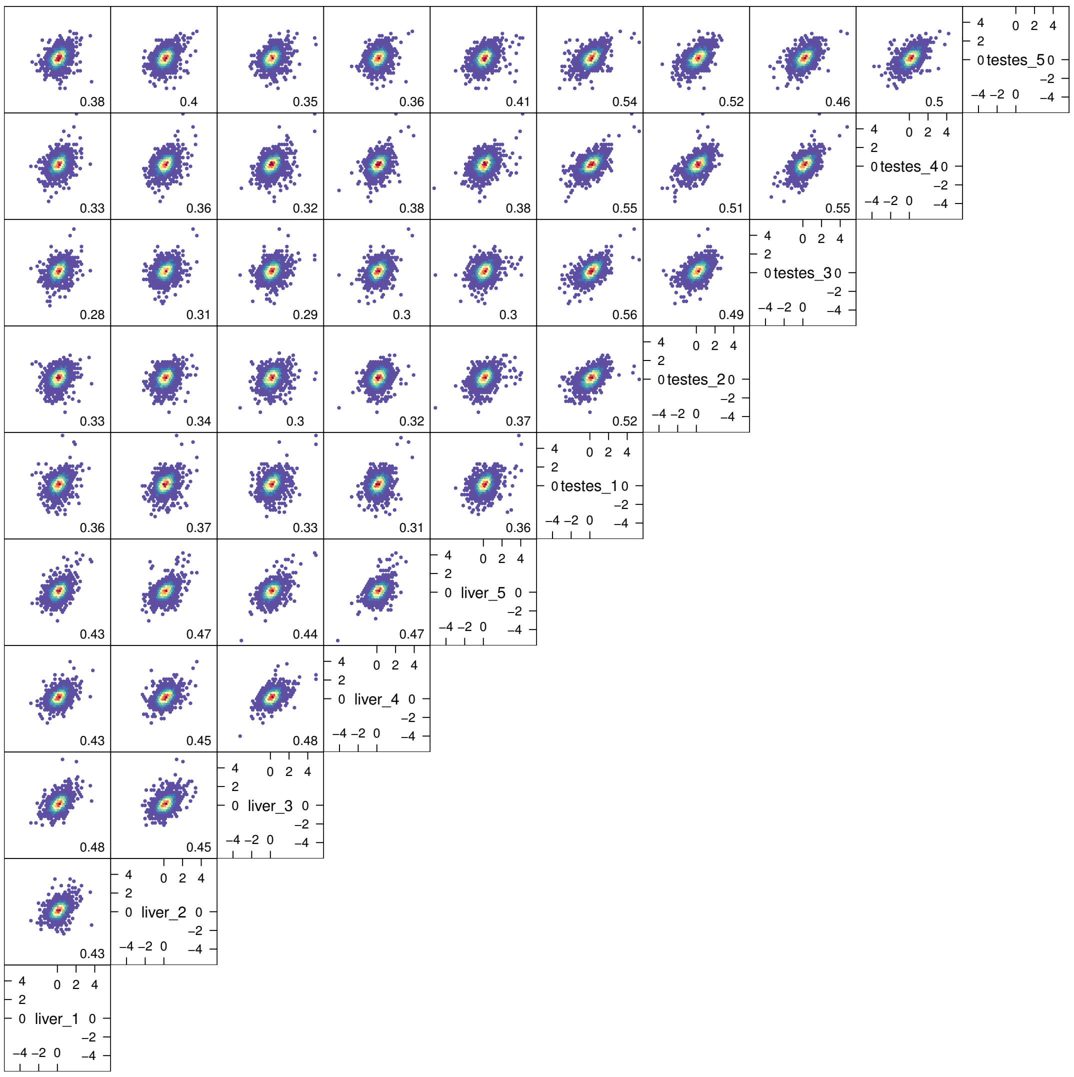


Figure 4.9: Comparison of k-mer importances for classifiers trained to detect liver NMIs versus classifiers trained to detect testes NMIs (in human). Classifiers were trained on experimentally determined NMIs from human liver or testes, and a SVM was trained to separate these regions from the rest of the genome. The input data was randomly split into training and test sets 5 times, and each replicate is shown. Points are binned and the density of points in a region is represented with red (high density), yellow (medium density) or blue (low density). The pairwise Pearson correlation coefficient is shown in the lower right corner of each plot.

4.4.2 Differentiating between testes and liver NMIs in six vertebrates

The previous section we showed that using a genome-wide classifier trained on one tissue, we were able to predict NMIs in another tissue with high accuracy. This was true across all six vertebrate species that were tested, and in both liver and testes. Additionally, comparing the most important features of these classifiers identified very few k-mers that had a consistent difference in importance in the two tissues. This suggests that the sequence features of NMIs in both tissues are similar, and that the tissue specific differences in methylation are not encoded or influenced by the DNA sequence of the NMIs. To test this, we took the NMIs that are uniquely present in either liver or testes, and tried to train a classifier to differentiate between the two sets of NMIs.

In this analysis we are most interested in more complex sequence features, so we decided to first check for some simple biases in the input sequences, and control for them. Specifically, we checked the length of the sequences, CpG ratio, and total number of sequences from each tissue. First, we compared the distributions of NMI lengths between the two tissues in all six organisms. This was done because length of the input sequences can bias the classifier, especially when using relatively short sequences and long k-mers (which results in a sparse vector of k-mer frequencies). This is regardless of the fact that the spectrum string kernel has the option to normalize the feature vector, which in my experience only works well when the feature vector is more dense. Next, we checked the CpG ratio distribution of the NMI sequences in each tissue. Looking at the resulting distributions (see Figures 4.10 and A.1), we see that the distributions of both lengths and CpG ratios are quite different in several of the organisms. To remove the influence of these factors from the analysis, we simply binned the sequences based on each of these variables, and downsampled the sequences in each bin so that there is a matching number of sequences in that bin from both tissues. Looking at the same distributions after performing this procedure (see Figures 4.10B and A.2), we see that the distributions are nearly identical for the two tissues. Additionally, this downsampling also leaves us with an identical number of sequences from each tissue, correcting for another potential bias.

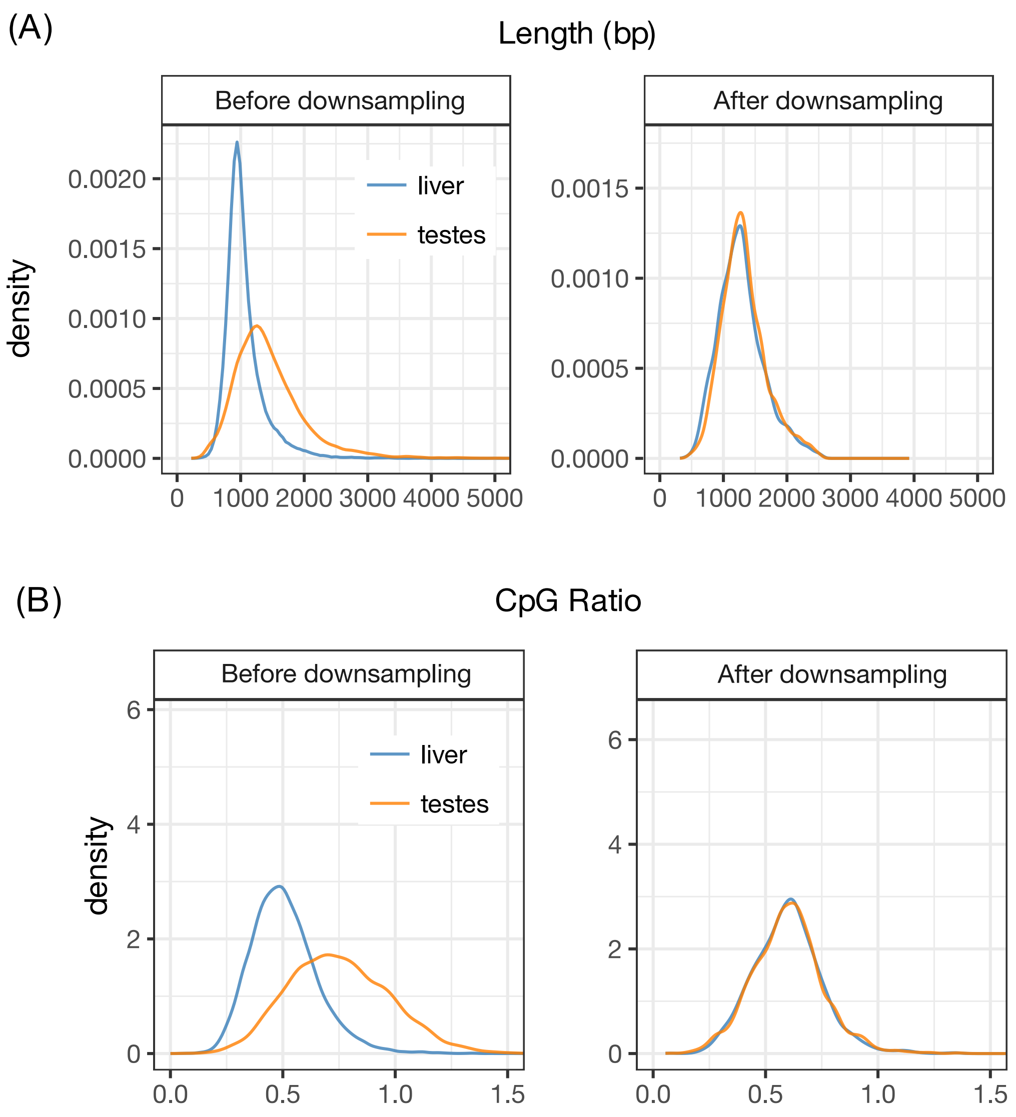


Figure 4.10: Zebrafish NMI length and CpG ratio distributions. (A) Length and (B) CpG ratio distributions of liver and testes NMIs in zebrafish before and after downsampling. For other organisms see Supplementary Figures A.1 and A.2 in Appendix A.

The remaining liver and testes NMI sequences were then used to evaluate whether or not their sequences can be used to identify which tissue the NMI is active in. The number of NMIs were also randomly downsampled so that the same number of sequences were used for each species, allowing us to more easily compare the results for different species. We then used nested cross-validation to estimate the performance of the classifier on new data (outer cross-validation loop), as well as choose the best k-mer length (inner cross-validation loop). The outer cross-validation was 10-fold cross-validation, and inner cross-validation was 5-fold cross-validation. For each of the 10 outer cross-validation steps, the optimal k-mer length was chosen using an inner run of 5-fold cross-validation, which was then used to retrain the model and predict on held out test data which is made up of 10% of the whole data set.

The nested parameter tuning over different k-mer lengths shows that the k-mer length that achieved the highest average AUROC in all organisms was 5-mers, with the exception of mouse which had a slightly higher average AUROC using 4-mers (see Figure 4.11). Overall the results show that it is possible to differentiate between liver and testes NMIs in all six species with high predictive performance (mean test set AUROC 0.70–0.89, see Figure 4.12), using only the DNA sequence of the NMI.

4.4.3 NMIs in 10 human tissues and cell types

The previous section has shown that across six different vertebrate species, DNA sequence can be used to predict whether a given NMI is present in liver or testes with high accuracy. While this is interesting, the analysis is limited by the fact that the Bio-CAP data we used is only available for two tissues in all six organisms that were studied. To determine if the results generalize to more tissues, it would be necessary to have experimentally determined NMIs for more cell types. Fortunately, Mendizabal and Yi (2015) collected whole genome bisulfite sequencing (WGBS) data from 10 human cell types and tissues, and also processed this data and defined NMI regions. First, we will compare the NMIs identified using Bio-CAP by Long et al. (2013) to those identified using WGBS by Mendizabal and Yi (2015), and then we will explore the predictive power of DNA sequence in

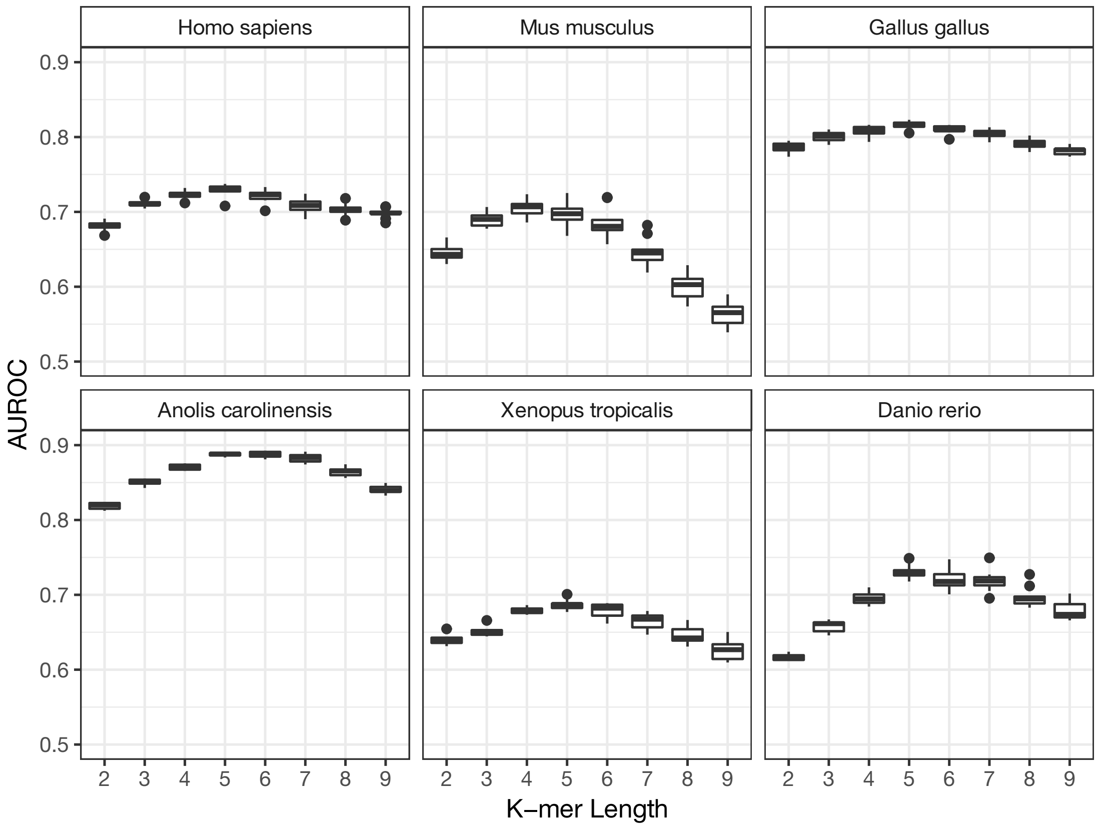


Figure 4.11: Parameter tuning for liver vs. testes NMI prediction. Grid search over k-mer lengths from 2-mers to 9-mers. The results show the average AUROC of 10 runs of 5-fold cross-validation.

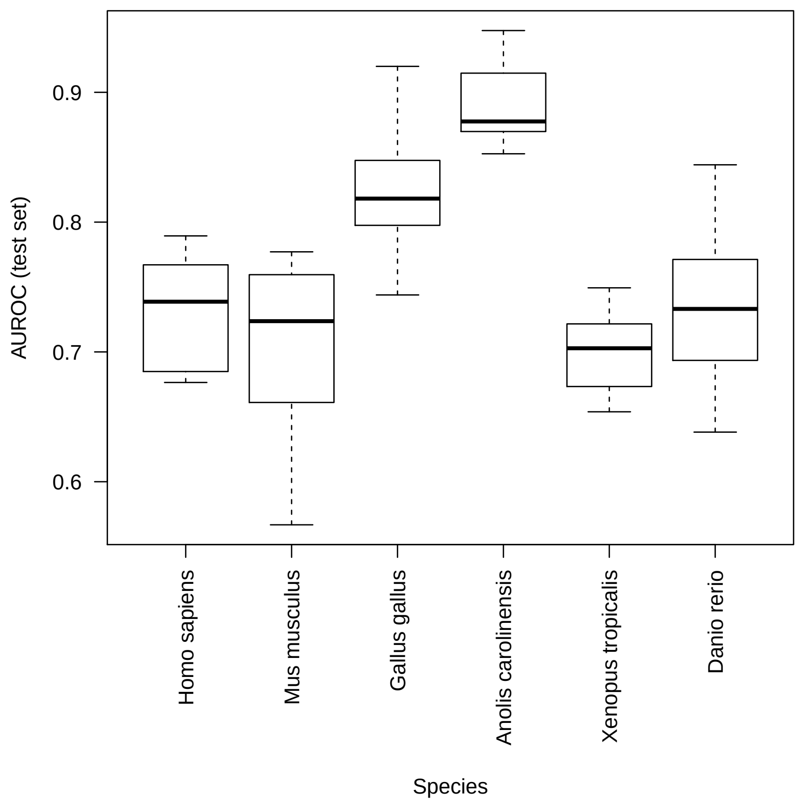


Figure 4.12: AUROC when predicting NMI liver vs. testes in six vertebrate species. 10-fold cross-validation using optimal parameters from an internal 5-fold cross-validation loop (see Figure 4.11).

WGBS-based NMIs across 10 human tissues.

4.4.4 Comparing Bio-CAP NMIs to whole genome bisulfite sequencing NMIs

Before starting to analyze the NMIs across all 10 cell types, we will first compare the NMIs that have been identified using the Bio-CAP method by Long et al. (2013) with those identified using WGBS data by Mendizabal and Yi (2015). As outlined in Section 2.3, the two experiments are fundamentally very different, and it is expected that this may result in differences in the non-methylated regions that are identified using each method.

To compare the two methods, we used the one tissue for which we have both Bio-CAP and WGBS data: human liver. As a first step, we used a genome browser to visually compare the experimental data (mapped reads for the Bio-CAP data, and beta values for the WGBS data), as well as the NMI regions that were called by the authors of each study using that data.

When looking at the broad trends in the experimental data, the two methods look like they identify very similar regions (Figure 4.13A). Overall, the Bio-CAP regions are more broad and the signal is smoother, but it is quite rare that there is a signal in one experiment and nothing in the other. Looking at several regions in more detail, the base pair resolution of the WGBS method becomes apparent, and it is also possible to see regions that seem to have very few CpGs. Also, the WGBS data seems to be relatively binary ($\beta = 0\%$ or 100% in most cases), while the Bio-CAP data varies more in signal intensity.

Comparing the actual regions that are called NMIs by each method, especially when zoomed in close to a single region, we observe quite a lot of variability. Bio-CAP NMIs look much more broad than WGBS NMIs, and possibly extend beyond the end of the true non-methylated region (Figure 4.13B–D). In contrast, the WGBS NMIs are very short, and seem to break up regions that should be a single continuous NMI into multiple pieces, missing out on part of the NMI region in the process (Figure 4.13D). Also, the criteria for calling a region an NMI with WGBS data seems to be more stringent, resulting in the method missing some regions that are called NMIs by the Bio-CAP method (Figure 4.13A right side). Overall the Bio-CAP NMIs look more consistent with its associated raw experimental data.

One explanation for this is that Bio-CAP NMIs are identified by identifying clusters of reads in the mapped sequencing data using the tool MACS, a computational method based on a rigorous statistical model that is commonly used to identify peaks in ChIP-seq data. In contrast, to call regions from the WGBS data Mendizabal and Yi (2015) used a more ad-hoc method, using a 200bp sliding window and a 50bp step size, they extended the window until it contained $< 80\%$ of sparse (< 0.2 beta value) methylation. Additionally, the windows had to contain at least 10 CpGs. This ad-hoc approach could clearly be improved.

The qualitative observations based on looking at a handful of regions in the genome browser were then evaluated quantitatively. A histogram of the widths of all the NMI regions called by each method confirms that the NMIs identified by Mendizabal and Yi (2015) tend to be much shorter than the NMIs identified by Long et al. (2013) (Figure 4.14). The most common length of the WGBS NMIs is exactly 200 bp, corresponding to the window size used by the authors when post processing the experimental data, and nearly all NMIs are less than 2 kb. In contrast, the Bio-CAP NMIs are much longer, with very few NMIs having a length less than 750 bp and the vast majority being between 1 kb and 6 kb.

To evaluate the effect of the potentially too broad Bio-CAP NMIs, we systematically removed increasing amounts of sequence centered on the borders of the Bio-CAP NMI regions, and then evaluated how well the left over regions can be predicted using the same whole-genome classification scheme as was discussed extensively in Section 4.4. The results show that predictive performance (AUROC and AUPRC) increases up until between 1 kb–2 kb of the NMI border regions are removed, and then the predictive performance drops again (Figure 4.15). While this isn't conclusive evidence that the NMIs called using the Bio-CAP method are approximately 1.5kb too broad in most cases, it does support the idea that the flanking regions at least contain different sequence patterns than the inner regions, most likely containing what have been described previously as CpG shores (Irizarry et al., 2009).

In summary, both experimental methods are in relatively high agreement. The Bio-CAP NMIs look like they extend beyond the region that is supported by the

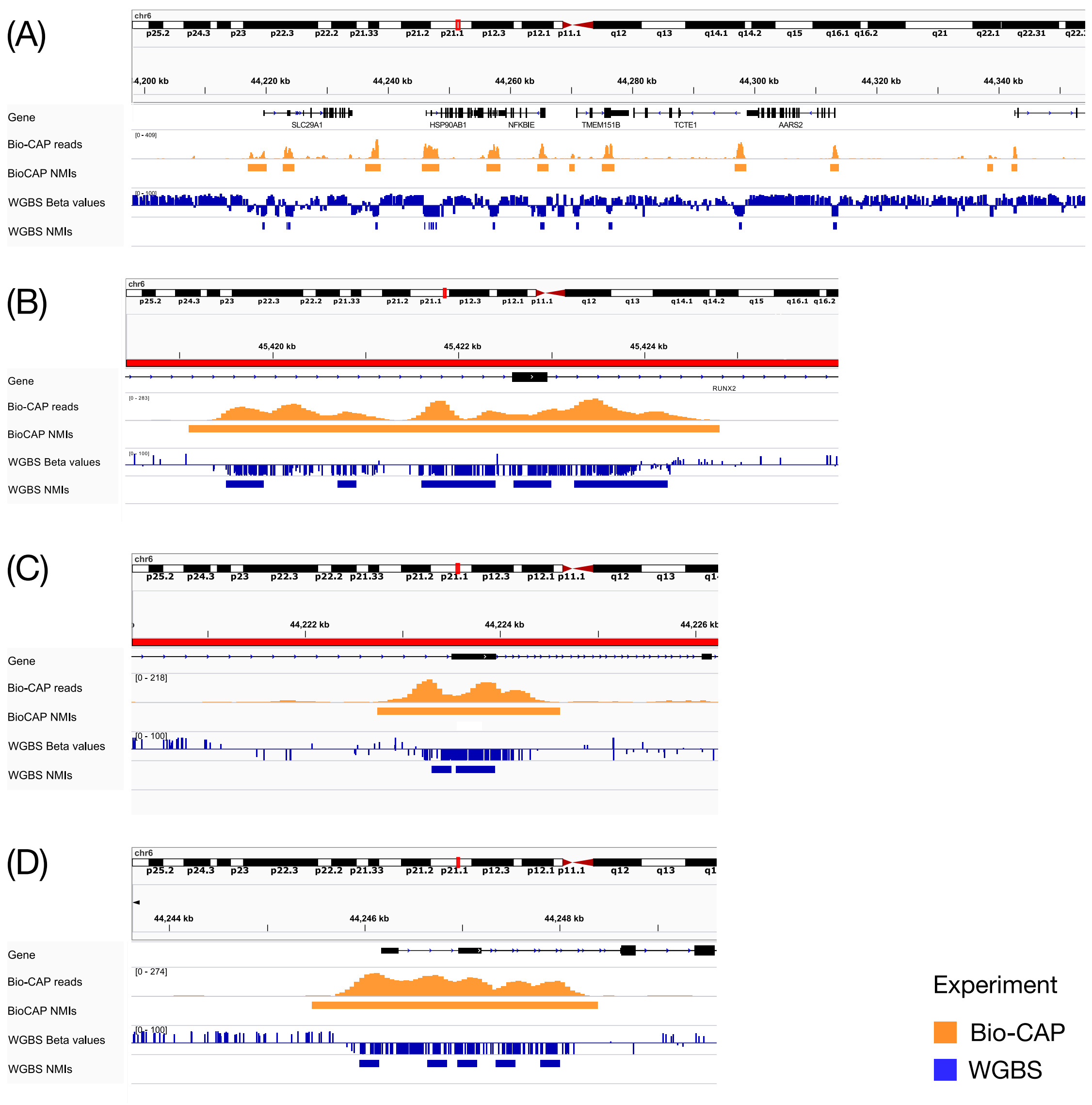


Figure 4.13: Example non-methylated genomic regions showing Bio-CAP and WGBS data, along with the NMIs that are called from this data. Bio-CAP data is shown in orange and WGBS data is shown in blue. There are two tracks for each experiment: raw experimental data as first track, and post processed NMIs as the second track. (A) A typical large genomic region showing qualitatively high agreement between the NMIs identified using Bio-CAP data and those identified using WGBS data. (B) A more detailed view of a single region, showing a single Bio-CAP NMI split into multiple WGBS NMIs. (C) The NMIs extend further beyond the core non-methylated region in Bio-CAP NMIs. (D) A region with very low methylation is missed by the method which calls NMI regions from WGBS data.

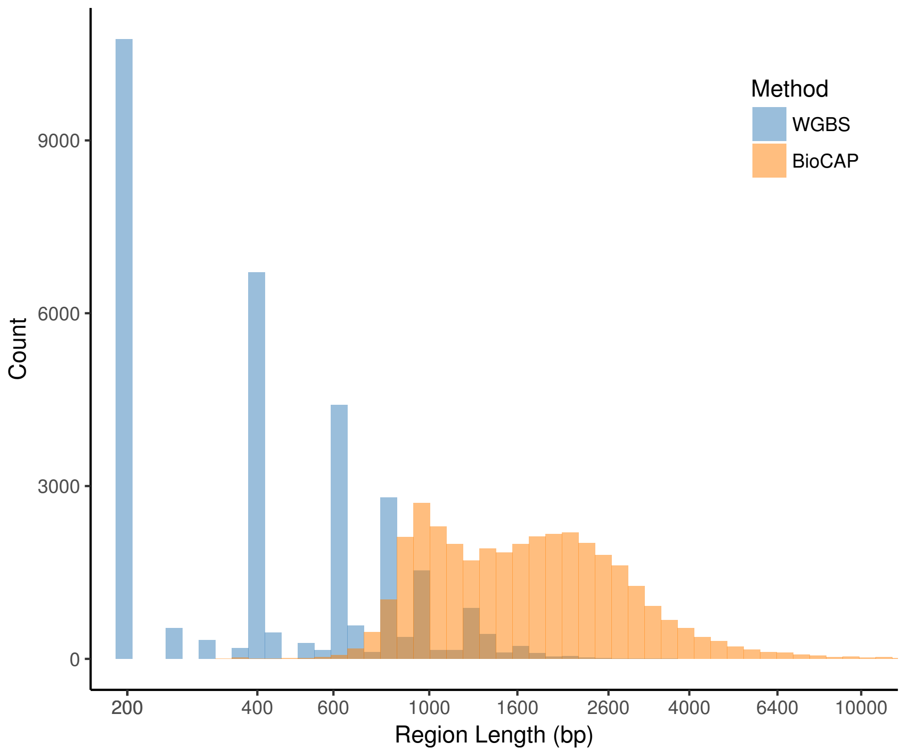


Figure 4.14: Histogram of NMI lengths. NMIs were identified using Bio-CAP and whole genome bisulfite sequencing experiments in human liver.

experimental data, and a region of approximately 1.5 kb centered on the borders of the NMIs should be excluded from a high confidence NMI training set. The WGBS experiment looks well suited for identifying NMIs, but the method for calling NMI regions from the experimental data breaks up many NMIs into smaller pieces, and misses out on parts of the NMI. We can say the WGBS NMI calling method has *high precision*, because the regions it calls NMIs look like they are strongly supported by the experimental data, but *lower sensitivity* because it misses out on some regions that the experimental data strongly suggests are non-methylated.

4.4.5 Predicting NMI presence in 10 human tissues

In light of the analysis in the previous section, we can be confident that the regions that are called WGBS NMIs by Mendizabal and Yi (2015) are indeed non-

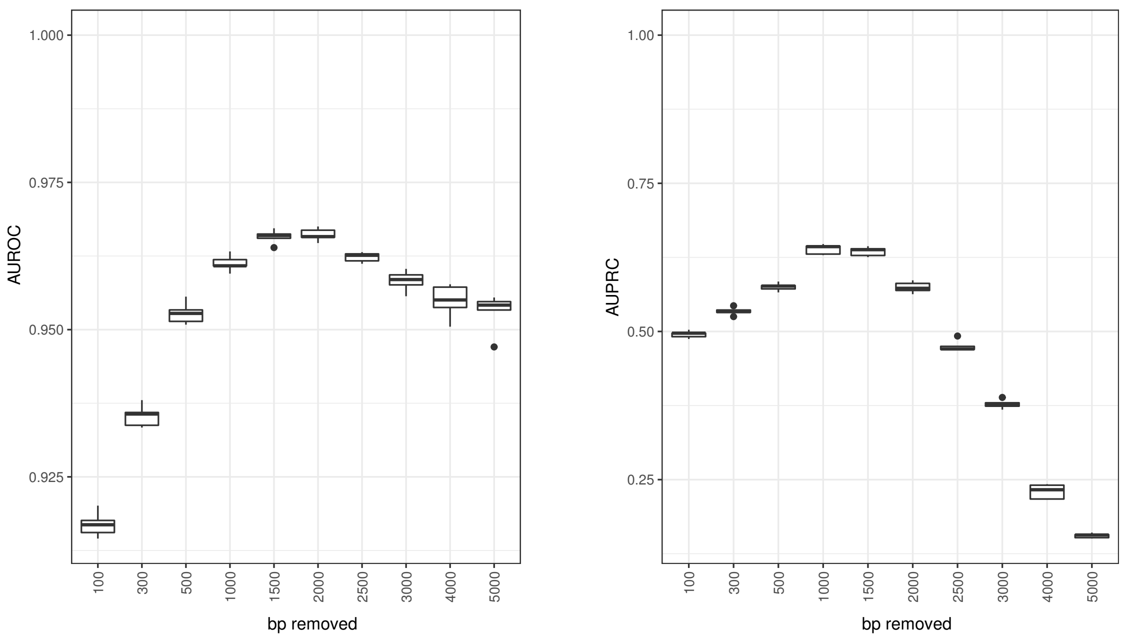


Figure 4.15: Predictive performance of successively narrower NMI regions. Whole-genome predictive performance across 5 random runs using Bio-CAP NMIs that have had regions around their borders removed from the analysis. Both the AUROC and AUPRC are shown, and in both cases the optimal predictive performance is with 1kb–2kb of border region removed.

methylated regions. We used them to directly compare WGBS-based NMIs in a wide range of tissues (10 human tissues), to determine if the tissue where a region is unmethylated can be predicted based on sequence. Because we are interested in comparing tissues, we have removed from the analysis all regions that are ubiquitously non-methylated in all tissues. The remaining 27,589 (out of an original 51,572) regions were then used as input to separate classifiers that were trained for each tissue. For these classifiers, the *positive set* is the NMIs that are non-methylated in the tissue of interest, and the *negative set* is the NMIs that are methylated in that tissue but are non-methylated in at least one other tissue. The number of NMIs present (non-methylated) in each tissue are shown in Figure 4.16.

Using 10-fold cross-validation on each tissue, we trained string kernel SVMs to try to predict whether a given DNA sequence belongs to an NMI that is non-

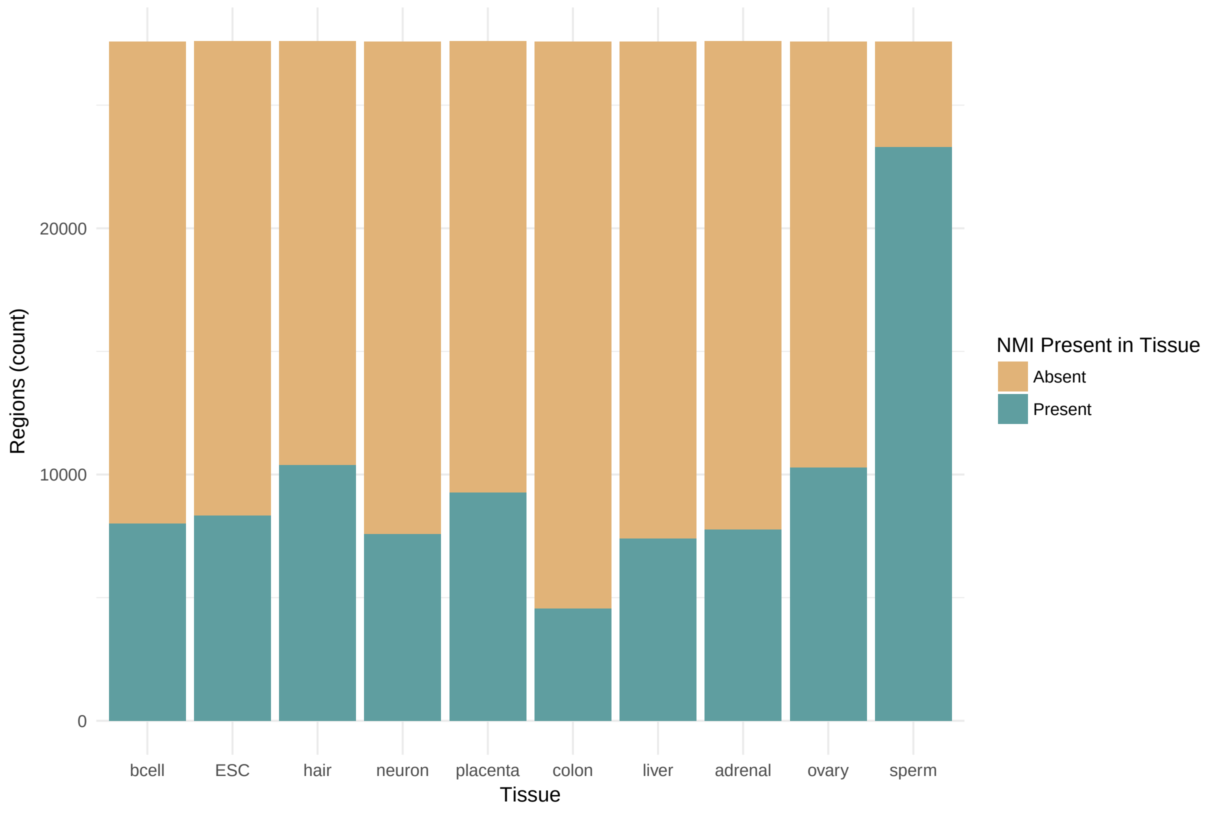


Figure 4.16: The number of WGBS-based NMIs in 10 human tissues. Only NMIs that are not ubiquitously present in all 10 tissues are shown. Ubiquitous NMIs were excluded from the entire analysis.

methylated in that tissue or not. For this analysis we evaluated different string kernels to see if we could improve predictive performance: the spectrum kernel (used throughout the earlier part of this thesis), the mismatch kernel, and the gappy pair kernel (see Section 3.2.2 for details).

As we see from the results (Figure 4.17), the choice of kernel has no consistent effect on predictive performance. For colon and placenta, the gappy pair kernel achieves a noticeable improvement in AUROC, but in all other tissues there is little difference between kernels.

Regardless, the average AUROC values show that in 8/10 tissues, it is possible to predict whether or not a region is non-methylated in a particular tissue with fairly high accuracy (average AUROC of approximately 0.8, see Figure 4.17). In the remaining two tissues, placenta and colon, the predictive performance was significantly lower (< 0.7 AUROC). To investigate if these two tissues would be expected to be outliers, we compared the correlation of NMI presence and absence across all regions in every pair of tissues, performed principal component analysis (PCA) on the matrix of NMIs, and compared the number of unique NMIs per tissue. This comparison did not yield any single indication as to why these two tissues are outliers. Placenta is an outlier in the PCA plot, but sperm is even more of an outlier (see Supplementary Figure A.3 in Appendix A). Colon has very few NMIs that are unique to that tissue (only 1), but liver and adrenal gland also have very few (19 and 9 respectively). Further analysis would therefore be required to explain the difference in predictive performance in these two tissues.

4.5 Discussion

Our results demonstrate that highly informative tissue-specific DNA sequence features are contained within experimentally determined non-methylated regions of DNA, which can be used to identify NMIs genome-wide in several vertebrates. Using known non-methylated regions of the genome to both train and evaluate our methods, we were able to show that DNA sequence can be used as a predictor for non-methylated regions in all six vertebrates that were examined, and that sequence can also be used to predict per-tissue NMI activity. This was not entirely expected because existing sequence-based CpG island predictions have a low overlap with true non-methylated regions, suggesting that DNA sequence might not be highly predictive of non-methylated regions (Long et al., 2013). The high performance of the SVM classifier in all organisms and tissues proves that despite the low overlap with existing CpG island predictions, the use of a more complex model with longer subsequences as features and a supervised learning approach demonstrates that there are DNA features in all vertebrates that are predictive of NMI regions.

NMIs in different species We show that using a string kernel support vector machine these informative DNA features can be used to predict the location of non-methylated islands genome-wide better than existing CpG island prediction

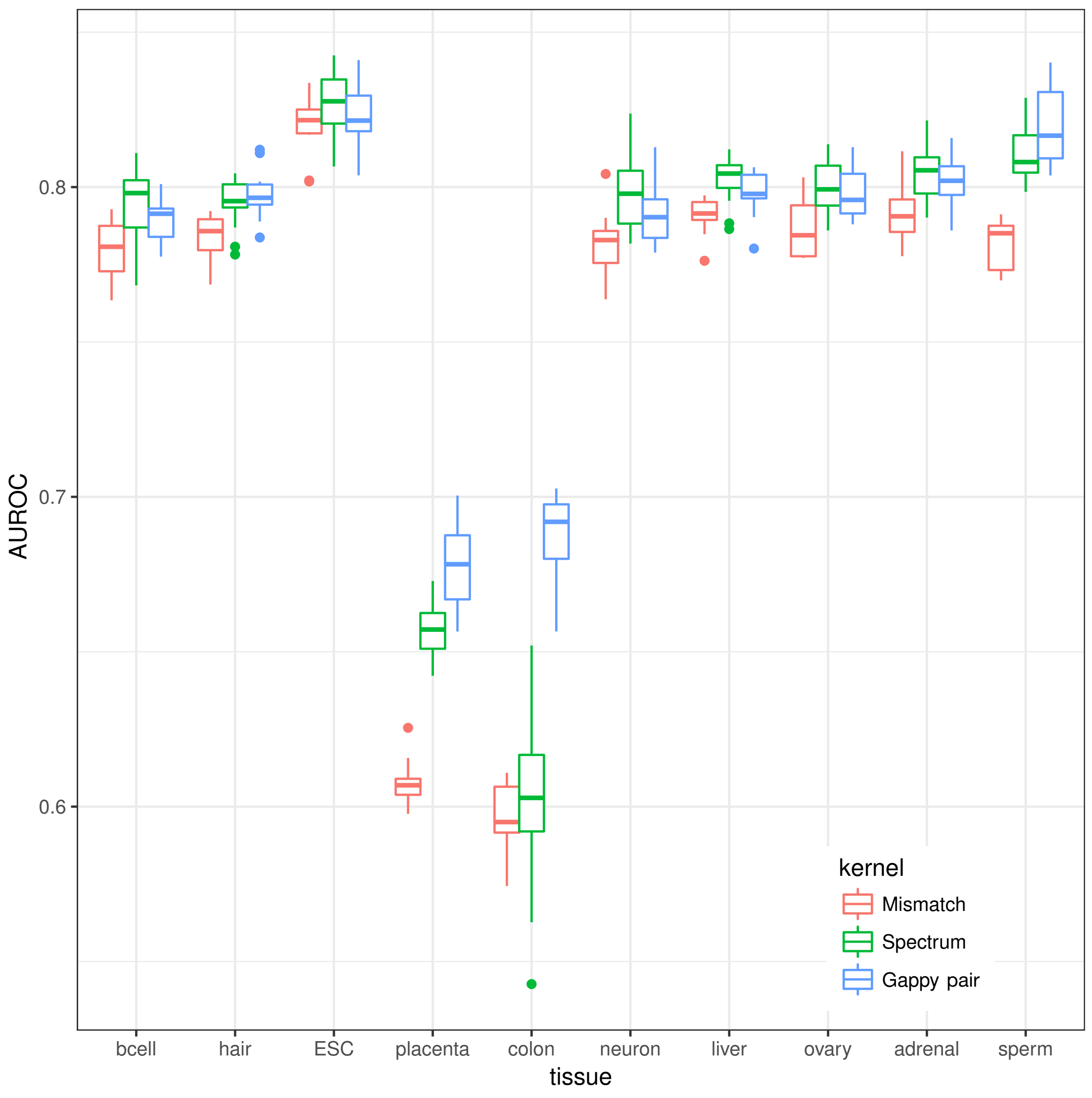


Figure 4.17: Classification performance of NMI tissue prediction in 10 human tissues. The classifiers were trained with a foreground consisting of all regions where an NMI was present in the given tissue, and the background set was the NMIs that are absent in that tissue but present in another tissue. Three different kernels were used: the spectrum kernel, mismatch kernel, and gappy pair kernel. The distribution shows the AUROC of the classifier across the 10 cross-validation folds.

methods. Given the fact that NMIs only make up 2–4% of the genome, only a classifier that can control false positives will be useful for predicting non-methylated regions genome-wide. The precision-recall curves show that for all organisms we are able to predict NMIs better than all existing methods, and that for cold-blooded vertebrates we are uniquely able to identify a modest number of non-methylated regions while maintaining a low false discovery rate.

Additionally, we have shown that NMI predictions made by training a classifier in one species and predicting in another can be highly accurate when performed between warm-blooded species and to a lesser extent lizard, while cross-species prediction in zebrafish and frog proved to be very difficult. One explanation for the differences we are observing in cross-species prediction accuracy is that NMI regions in all six vertebrates are identified based on two factors: first, simple CpG-richness, and second, more complex sequence features (potentially including transcription factor binding sites) that are species and/or tissue-specific. These two factors have a different level of importance in each organism. In warm-blooded vertebrates, CpG richness is the most important factor by far, while in cold-blooded vertebrates more complex sequence features are more important. This is why a simple classifier using dinucleotide frequencies works so well in warm-blooded vertebrates, and why this simple classifier is easily transferable to other warm-blooded vertebrates. In contrast, in zebrafish and frog CpG content still contributes to the prediction of NMIs, but the contribution of more complex sequence features is more important. These complex sequence features are species-specific, and therefore do not transfer well between species, resulting in poor cross-species prediction accuracy in zebrafish and frog. Lastly, the importance of the two factors in lizard is more balanced. CpG content is more important than in other cold-blooded vertebrates, but complex sequence features are more important than in warm-blooded vertebrates. This explains why we observe fairly good prediction of NMIs in lizard after training in warm-blooded vertebrates, which are identifying the simple CpG richness that is somewhat important in lizard NMIs. On the other hand, the classifiers trained in other cold-blooded vertebrates have learned to identify complex species-specific sequence features and place less importance on CpG richness, and therefore perform poorly at predicting lizard NMIs.

The important k-mers for genome-wide prediction identified by the SVM did show clear differences between warm-blooded and cold-blooded organisms though, with the highest scoring k-mers containing almost no A/T nucleotides in warm-blooded organisms, and nearly every high scoring k-mer containing an A/T nucleotide in cold-blooded organisms (see Table in 4.5). This disagreement between the importance of the k-mers, which suggests a role for sequence-specific transcription factors, and the lack of known JASPAR motifs could suggest that the factors involved in establishing and maintaining NMIs in cold-blooded organisms may have changed along with composition of cold-blooded NMI sequences themselves. And because of this our databases do not adequately reflect those transcription factor binding sites in cold-blooded vertebrates.

These findings are complemented by those presented in a recent paper in which the authors experimentally evaluated whether stretches of genomic sequence containing NMIs from one organism can also evade methylation when placed into another organism (Long et al., 2016). Their study demonstrated that transplanting a segment of genomic DNA from human into mouse, or from mouse into zebrafish, resulted in the majority of NMIs within these regions remaining unmethylated. In particular, these findings show that the fish is capable of protecting mouse CpG islands from being methylated. Nevertheless, the typical fish NMI sequence has evolved in the direction of the sequence patterns that we identified in this analysis. This again raises the point that there should be a particular functional pressure that has led it to do so, for example transcription factors whose motifs are not currently known in cold-blooded vertebrates.

NMIs in different tissues We additionally showed that the NMIs DNA sequence is not only predictive of whether or not that region will be methylated, but also in which tissues it will be methylated or unmethylated.

For liver and testes tissues in six vertebrate species, we have shown that training an NMI on one tissue and predicting in another does not greatly decrease the predictive accuracy of the classifier in comparison to training and predicting on the same tissue. This is in spite of a large set of tissue-specific NMI regions that exist in each species.

Nevertheless, we show that a string kernel support vector machine is able to predict which tissue an NMI is active in based on its DNA sequence with high accuracy (AUROC ≈ 0.8). This was demonstrated for two tissues in six vertebrate species, and for ten tissues in human. In human tissues, colon and placenta NMIs were noticeably more difficult to predict than those from the rest of the tissues. There was not a clear technical reason for the difference, so it is possible that the difference is biological. In comparison to the other tissues that were used in the study, placenta is a relatively heterogeneous mixture of cell types which may have an affect on the analysis. Colon may have more variable methylation and be more easily altered by environmental and dietary factors, just as lung tissue demonstrates changes in DNA methylation in smokers (Zeilinger et al., 2013). However these hypotheses would need to be tested in a later study.

Chapter 5

Predicting enhancers using a small subset of high confidence examples and co-training

In this chapter we present the results of taking a conceptually different approach to the computational prediction of regulatory regions of the genome called enhancers. First, we take a more conservative approach to defining our training set by requiring that the regions in the training set are identified by three completely different experimental methods (CAGE, HiCAP and a classical enhancer reporter assay, see Section 2.4 for details). Second, because this conservative approach yields a small training set, rather than using the standard method of supervised learning we chose to leverage both labeled data from our training set and unlabeled data using a semi-supervised learning approach (see Section 3.1.2). Specifically, we use a variant of semi-supervised learning called co-training, which allows us to make our predictions based on multiple sets of input features, in this case the abundance of histone modifications as one set and the DNA sequence as another set.

The content of this chapter is adapted from a paper that was co-first authored with Anna Ramisch, and was presented at the German Conference on Bioinformatics 2016. Anna Ramisch did the initial implementation of the algorithm, data preprocessing, and exploratory plots, and I did extensive rewriting of the algorithm, methods for parameter tuning, summary plots and the overfitting analysis. We both worked on the paper along with the other co-authors, a preprint of which is available at PeerJ Preprints (Huska et al., 2016).

5.1 Motivation

Given the importance of enhancers in the context of gene regulation and disease, it is critical to be able to identify them in the genome. A multitude of experimental and computational methods have been developed over the last several years with the goal of genome-wide enhancer identification. Originally, cross-species genomic sequence conservation was used to identify enhancers, with the assumption that important regulatory elements would be conserved through evolution (Pennacchio and Rubin, 2001; Nobrega, 2003; Pennacchio et al., 2006; Visel, Bristow, and Pennacchio, 2007; Visel et al., 2008). However, experimental validation of highly conserved regions using either *in vivo* or *in vitro* reporter assays showed that only approximately half of them were able to act as enhancers (Pennacchio et al., 2006; Visel et al., 2008). Later, methods were developed that could identify DNA-protein interactions on a genome-wide scale, including chromatin immunoprecipitation followed by microarray (ChIP-chip) and chromatin immunoprecipitation followed by high-throughput sequencing (ChIP-seq). These methods were used to try to identify enhancers based on the relationship between certain modified histones (mainly H3K4me1 and H3K27ac) or histone modifying proteins (such as p300).

More recently, methods for the large-scale identification of DNA-DNA interactions have been used for enhancer prediction as well (Sahlén et al., 2015; Chepelev et al., 2012; Li et al., 2012; Rao et al., 2014). This is based upon experimental evidence that enhancers function by physically looping to interact with the regions that they regulate (Müeller-Storm, Sogo, and Schaffner, 1989). Another method, STARR-seq (Arnold et al., 2013), in essence performs a large-scale version of a classic enhancer reporter assay that can measure ectopic enhancer activity, has been able to identify genomic regions with enhancer potential genome-wide in drosophila, and in selected regions in humans. Lastly, enhancers were recently hypothesized to initiate bidirectional RNA polymerase II (RNAPII) transcription, producing so-called eRNAs (Kim et al., 2010). Based on CAGE experiments from the FANTOM5 consortium, consisting of data from hundreds of cell lines and tissues, Andersson et al. (2014) identified more than 40,000 putative enhancer regions, together with their activation levels across human tissues, marked by the presence of bidirectional capped transcripts.

5.1.1 Computational methods for enhancer prediction

To complement the experimental methods, computational methods have been developed that use data from some of the preceding experimental methods as input in order to make more accurate enhancer predictions. Currently available computational methods can be divided into two broad classes of approaches: unsupervised methods, for example the genome segmentation algorithms Segway, ChromHMM and EpicSeg (Hoffman et al., 2012; Ernst and Kellis, 2012; Mammana and Chung, 2015), and supervised enhancer prediction methods such as RFECS (Rajagopal et al., 2013), EnhancerFinder (Erwin et al., 2013), and an SVM-based method by Lee, Karchin, and Beer (2011) (see Section 3.1.2 for a description of supervised and unsupervised learning).

The unsupervised methods do not rely on any knowledge about already identified enhancer regions, but extract patterns (for instance, of different chromatin states) directly from the data – an advantage when no experimentally validated information is available. While this sounds like an advantage, the downside is that we do have knowledge of certain regions in the genome to be actual enhancers, but unsupervised methods do not take advantage of this information.

The supervised methods on the other hand rely critically on the existence of a large high-confidence labeled training set of known enhancer and non-enhancer regions. Most supervised computational methods select a single set of experimental data, often created using one of the experimental methods mentioned above, and use it as their large labeled training set. This is not ideal because each experimental method only tests for one of the properties of enhancers that is currently believed to be necessary for their function: HiCap tests for looping, but regions can form loops without being active. ChIP-seq tests for the presence of certain histone modifications that are thought to be correlated with enhancer activity, but it is still unknown if this relationship is causal and the mechanisms for this relationship are still unknown. STARR-seq tests to see if the region can drive expression in a reporter, but this activity is ectopic and outside of the enhancer's native environment. The bidirectional transcription that Andersson et al. (2014) use to identify putative enhancer regions could be a mark of active enhancers (Andersson et al., 2014; Li, Notani, and Rosenfeld, 2016), or simply a mark of accessible chromatin (Young et al., 2016). Beside the fact that these methods tend to test for only one property of enhancer activity, they are also large scale methods which each have their own biases, whether that is sequencing, amplification or other technical biases. This means that the putative enhancers identified by any of the previously mentioned methods are likely to contain a number of false positives that make them a poor starting point for supervised learning.

In contrast to all the computational methods which rely on only one type of experimental method to define their enhancers, we choose not to fully rely on the results of a single type of experiment. Instead, we make the assumption that a region that is identified by several experimental methods is more likely to be a true active enhancer, because it has several of the properties of enhancer regions: the ability to loop to a promoter or other enhancer, the ability to boost the expression of nearby genes, a chromatin environment that is thought to be conducive to enhancer activity, etc. Therefore, we decided to start our enhancer prediction with a set of regions that are identified using not one but three separate experimental methods: HiCap, CAGE and ectopic enhancer assays as found in the Vista Enhancer Atlas.

This approach of choosing such a stringent set of enhancers when training a learning algorithm also has its disadvantages, in this case the size of the overlap between the enhancers predicted by all three methods is only 21 enhancer regions (see Figure 5.1). These are likely not enough examples for a fully supervised approach, so instead we choose a method which is a compromise between supervised learning and unsupervised learning: semi-supervised learning, specifically a variant of semi-supervised learning called *co-training* (see Section 3.1.2).

In this analysis we apply co-training to the problem of enhancer prediction because it allows us to start from a very small but high-confidence training set, and because we can incorporate sequence features and epigenetic features separately in order to improve prediction performance. For sequence features, we use dinucleotide frequencies of each genomic region. The epigenetic features are input-normalized ChIP-seq counts for several histone marks, transcription factors and histone modifying proteins. We focus on mouse embryonic stem cells (mESCs) because it is one of the few cell types for which sufficient experimental data is available.

5.2 Methods

5.2.1 Experimentally determined enhancers

In order to obtain our high confidence set of mouse enhancers that are active in embryonic stem cells with which to start training our model, we combined data from several sources which attempt to identify enhancers using different experimental methods. Each method tries to measure one property of enhancers, separately identifying regions that can *loop to promoters or other cis-regulatory regions*, are *bidirectionally transcribed*, and can *boost transcription* of a nearby gene in an enhancer reporter assay. The experiments that were used to identify these regions are all described in Chapter 2, and the specific datasets are outlined below.

Can loop to other regions: HiCap enhancers. We downloaded putative enhancers in mESCs from the set of high-resolution, genome-wide promoter-enhancer and enhancer-enhancer interactions determined with the HiCap technique by Sahlén et al. (2015). The set of putative HiCap enhancers comprises 71,698 unique genomic regions, which can be involved in more than one interaction.

Produces bidirectional transcripts: FANTOM5 enhancers. From the FANTOM5 Transcribed Enhancer Atlas (`http://fantom.gsc.riken.jp/5/datafiles/latest/extra/Enhancers/`) we downloaded 44,150 putative enhancer regions. These enhancers were defined by detecting bidirectional transcription at sites of enriched CAGE signal in various mouse cell lines and tissues (Andersson et al., 2014). The entire mouse permissive enhancers "phase 1 and 2" set was used.

Boosts transcription of a gene: VISTA enhancers. We downloaded 323 putative enhancer regions in embryonic mouse tissues from the VISTA Enhancer Browser (Visel et al., 2007). These regions, initially identified by extreme evolutionary sequence conservation or by ChIP-seq, were experimentally validated in a transgenic mouse enhancer reporter assay.

5.2.2 Co-training

Given the strict requirements for our training data, we decided to use a semi-supervised machine learning method so that we can take incorporate both labeled and unlabeled data into the learning process. Specifically, we used a variant of semi-supervised learning called co-training (Blum and Mitchell, 1998, see Chapter 3 for details). We compared that method to self-training, a more simple semi-supervised learning method, as well as supervised learning. For this work we used logistic regression as the base classifier, and as the supervised learning method as well.

5.2.3 Predictive Features

Because we are using the co-training method, we can use multiple separate sets of features as input to our predictive model. In this case, we chose to use ChIP-seq data as one feature set, since there are known correlations between certain histone modifications and enhancer activity. These associations seem to be the same regardless of cell type. As a second feature set, we used DNA sequence, because enhancer regions tend to be bound by cell type-specific transcription factors and therefore would be expected to contain binding sites for those transcription factors, or at the very least certain sequence biases. These DNA sequence features can potentially add some cell type-specificity to the more general cell type-independent ChIP-seq features.

ChIP-seq data The following mESC ChIP-seq datasets were used for building the first active enhancer classifier: H3K4me1, H3K4me2, H3K4me3, H3K27ac, H3K27me3, H3K36me3, H2AZ and corresponding input (GEO series GSE36114, E14 Day0 samples); and Pol2, p300, CTCF and corresponding input (GEO series GSE29184, samples GSM723019, GSM723018, GSM723015 and GSM723020). The data were pre-processed as described in Juan et al., 2016. Briefly, sra files were transformed into fastq files with sra-toolkit (v2.1.12) and aligned to the reference mm9/NCBI37 genome with BWA v0.5.9-r16 (Li and Durbin, 2009) allowing zero to one mismatches. We counted the overlapping ChIP-seq reads in the genomic regions of interest, namely enhancers, promoters and intergenic regions using bamsignals v1.2.1 (Mammana and Helmuth, 2014). Read counts in genomic regions where normalized by subtracting the log transformed input read counts by the log transformed ChIP-seq counts.

Genome Sequence We used the mouse genome as provided by UCSC (mm9, Jul. 2007), and calculated k-mer frequencies from these sequences to use as input features for the classifier. This was done using the `oligonucleotideFrequency()` function from the Biostrings v2.38.4 package in R v3.2.3.

5.2.4 Partitioning of the dataset

The entire dataset was partitioned into three distinct sets: a high confidence labeled set L, a hold-out validation set and an unlabeled set U. Each genomic region can only belong to one of these sets. The labeled set L and the unlabeled set U change their composition during the iterative procedure in the co-training and self-training algorithms. Note that in the following sections, regions are considered to be intersecting if they overlap by at least one base pair.

Initial labeled set The initial training set L is an almost fully balanced set of 21 positive and 28 negative examples. More precisely, the positive set is the intersection of enhancers defined by VISTA, FANTOM5 and HiCap experiments (as described above), and as such represents high-confidence enhancer regions. The negative set is composed of 14 promoter regions and 14 intergenic regions of size 300 bps, which were randomly chosen from the rest of the genome.

Validation set Our validation set consists of 500 positive and 500 negative examples. We chose positive examples for our validation set randomly from intersections of only two of the three putative enhancer sets (see Figure 5.1). The negative set is built from 250 randomly chosen promoter regions and 250 randomly chosen regions of size 300 bps from the rest of the genome.

Initial unlabeled set The unlabeled set U consists of 111,183 identified putative enhancers from VISTA, FANTOM5 and HiCap experiments, 21,359 protein coding gene promoters and 99,736 randomly chosen intergenic regions.

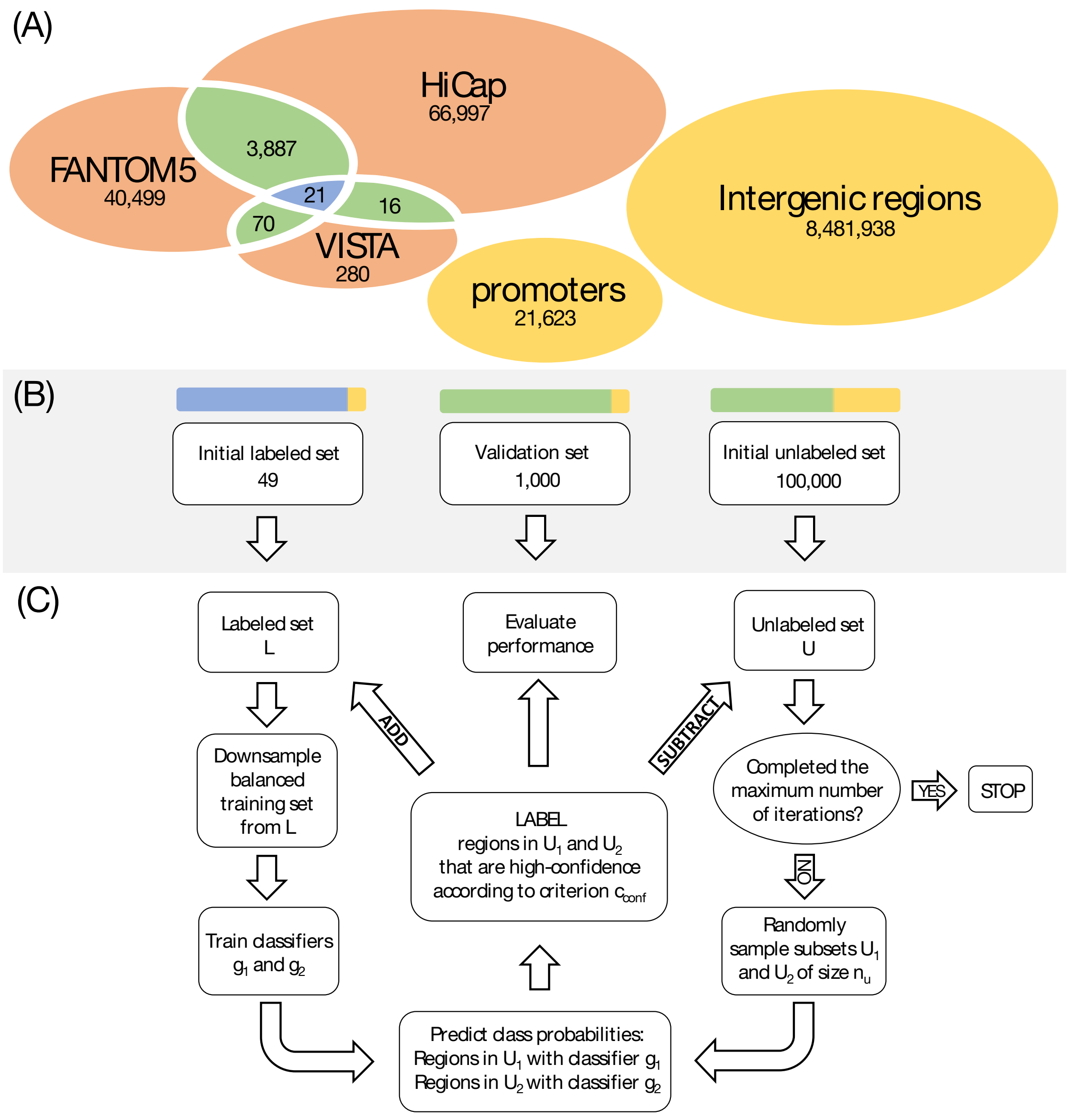

(A)
HiCap
66,997
FANTOM5
40,499
3,887
21
16
70
VISTA
280
Intergenic regions
8,481,938
promoters
21,623
(B)
Initial labeled set
49
Validation set
1,000
Initial unlabeled set
100,000
(C)
Labeled set
L
Evaluate performance
Unlabeled set
U
Downsample balanced training set from L
ADD
SUBTRACT
Completed the maximum number of iterations?
YES
STOP
NO
LABEL
regions in U1 and U2 that are high-confidence according to criterion cconf
Train classifiers g1 and g2
Randomly sample subsets U1 and U2 of size nu
Predict class probabilities:
Regions in U1 with classifier g1
Regions in U2 with classifier g2

Figure 5.1: Input data and workflow of co-training. (A) Venn diagram of putative enhancer sets, promoters and intergenic regions in the mouse genome. The intersection of all the putative enhancer sets includes 21 "high-confidence" regions. The pairwise intersection (in green) was used as a validation set. (B) Size and composition of the validation set, and the initial labeled and unlabeled sets. The colors in the bars refer to the genomic regions contained in the sets. (C) Workflow of our co-training method.

5.2.5 Parameter tuning

Both the co-training and self-training algorithms have several important parameters that need to be chosen. The parameters we focused on were (i) the size n_u of the unlabeled sets U_1 and U_2 that are sampled at each iteration step, and (ii) the criterion c_{conf} for defining the most confidently labeled examples that are added to the training set after each iteration.

In order to select good parameters for each method, we tuned these parameters using a grid search and selected the parameter combination that resulted in the highest average AUROC across four randomly chosen validation sets, with four random seeds per validation set, resulting in 16 runs of each parameter set total. The parameters that were tested were $n_u = 100, 200, 500, 5000$, and two criteria for selecting the most confidently labeled examples were evaluated. The first criterion was to just take a fixed number of examples per iteration, 6, 10, 25, 50, or 100 examples, and add them to the labeled set L. An additional parameter for this criterion was whether the selected examples should be forced to be equally split between positive and negative examples, or if the highest confidence examples are selected regardless of which class they belong to. The second criterion for choosing the most confidently labeled examples was using a score cutoff. Logistic regression gives scores between 0 and 1, so we evaluated using a score cutoff $s = 0.05$, 0.1, or 0.25, and all regions with a predicted probability $p \leq s$ or $p \geq 1 - s$ were added to the labeled set L. The optimal parameters for each method and the resulting AUROC is show in Table 5.1.

5.2.6 Experiments

Experiments were conducted to compare the methods against each other using different feature sets comprised of combinations of ChIP-seq and DNA sequence features. In detail, we compared co-training using ChIP-seq and DNA sequence feature sets to the following methods and feature sets:

- Semi-supervised self-training based on logistic regression with:
  - ChIP-seq-based features
  - Sequence-based features

  - All features (ChIP-seq and sequence-based)

- Supervised logistic regression with:
  - ChIP-seq-based features
  - Sequence-based features
  - All features (ChIP-seq and sequence-based)

5.3 Results

5.3.1 Very few enhancers are identified by all three experimental methods

In this work we first inspected the overlap between the three sets of experimentally determined enhancers that we are using to define our high confidence enhancers: FANTOM5 CAGE-based enhancers, HiCap-based looping enhancers and experimentally validated enhancers from the VISTA Enhancer Browser. The three datasets contain 44,150, 71,698 and 323 putative enhancers respectively, but the intersection between the three sets is only 21 regions as can be seen in Figure 5.1. Such a small intersection indicates that it is a non-trivial task to design a training set for a supervised enhancer classification task. Most other methods choose to use one of the three putative enhancer sets to train the model. In this scenario, the training set is large and fully supervised learning methods are expected to have a good performance, but it also means adding more noise and technique-specific bias. Another option is to choose the small subset of 21 enhancers at the intersection as a positive set. Given that enhancers at the intersection are supported by three completely different experimental techniques, we opted for the latter choice and adapted our learning methods to the smaller training data set by using co-training.

5.3.2 Optimal parameters for each method

By performing the parameter tuning as outlined in the Methods section, the optimal parameters for each method were determined. These values, along with the mean AUROC, are contained in Table 5.1.

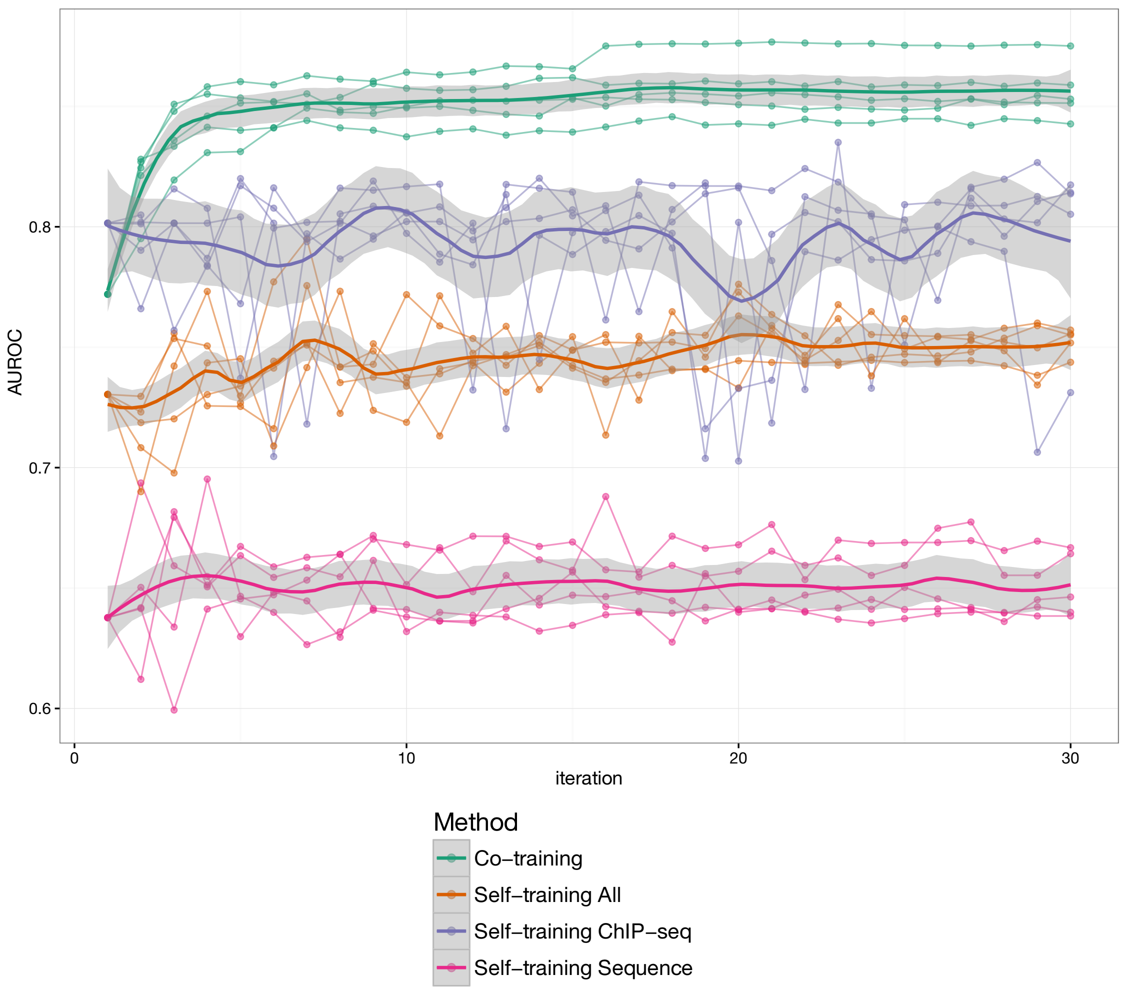


Figure 5.2: AUROC by training iteration. The performance (AUROC) of each classifier on one of the held out validation sets after each iteration of training. The performance measure used is the area under the receiver operating characteristic curve. Each method was run five times with a different random seed each time, which affects the random sample of unlabeled examples that is considered at each iteration. LOESS regression was used to plot a smoothed estimate for each method and the 95% confidence interval around these estimates is shown in grey.

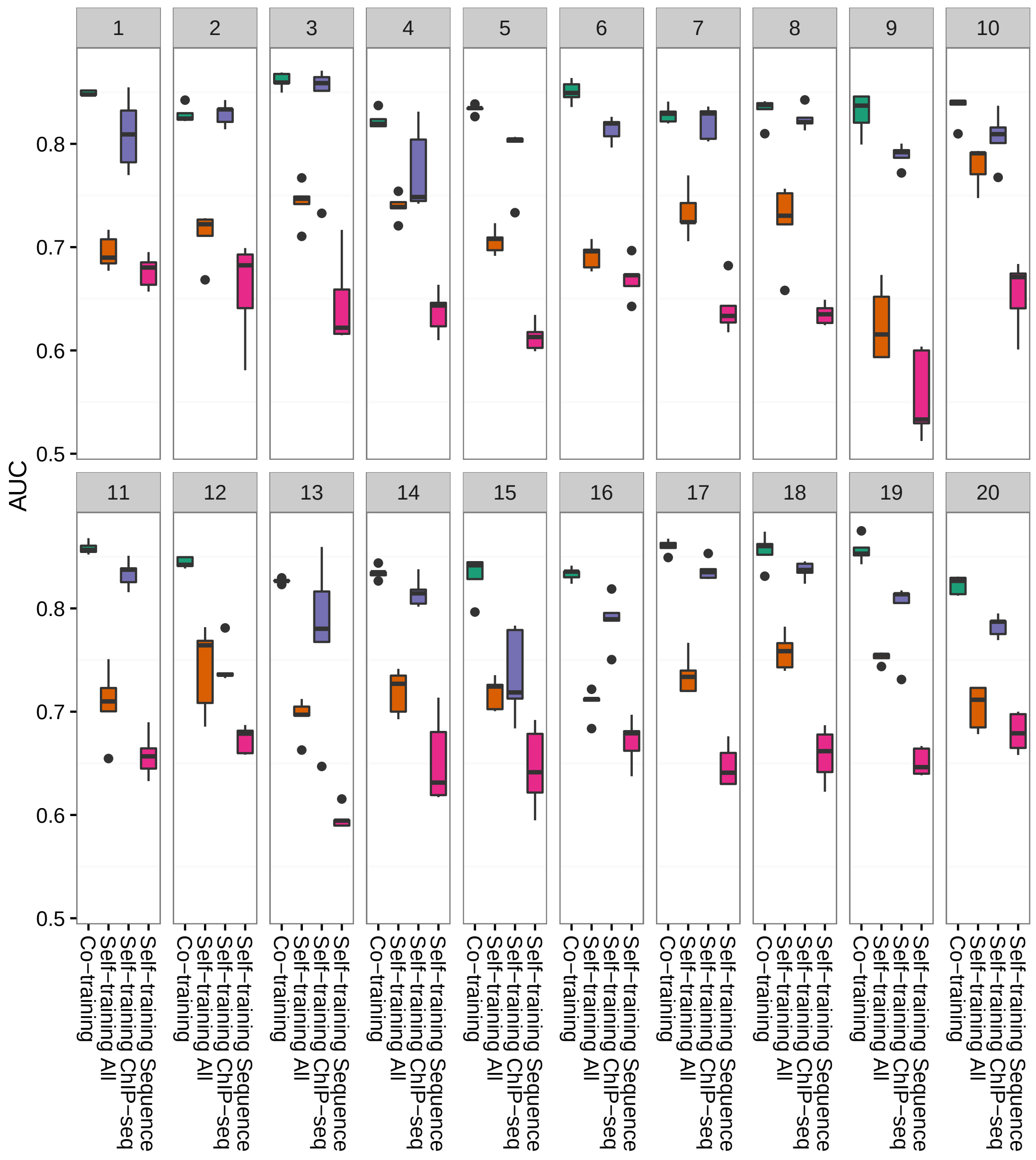


Figure 5.3: AUROC for 20 different validation sets. To investigate how sensitive the results are to the choice of validation set, we ran all methods on 20 different validation sets each comprised of a randomly selected 500 positive regions and 500 negative regions made up of 250 promoters and 250 other genomic regions. For each validation set, each method was run 5 times using a different random seed.

Table 5.1: Optimal Parameters For Each Method

Method and Features	n_u	c_{conf}	mean AUROC
Co-training	100	top 50 (force balanced)	0.84
Self-training All	5000	top 6	0.71
Self-training ChIP-seq	500	top 6	0.80
Self-training Sequence	5000	top 6 (force balanced)	0.68

5.3.3 Co-training outperforms other methods

Using the best set of parameters for each method as identified using the procedure outlined in the previous section we monitored the performance of each method at each iteration on 20 different validation sets (for one example, see Figure 5.2). In every case, and independent of the choice of the validation set, co-training performs on average better than the other methods (see Figure 5.3). In addition, unlike the other methods the classification performance (AUROC) reliably improves after incorporating unlabeled examples into the prediction.

5.3.4 Co-training greatly reduces overfitting

Looking at the change in performance over several training iterations (Figure 5.2), we see that in contrast to co-training, the self-training methods do not consistently improve in predictive performance as they iteratively add new examples to their labeled sets. One potential reason for this is that the self-training methods are overfitting, and that the additional examples they are adding to their labeled sets are not alleviating this problem.

One way to test for overfitting is to compare the predictive performance on the same data that the method was trained on, and compare that to the predictive performance on the held out test set. If the performance on the training set is much better than the performance on the test set, then it is an indication of overfitting. This test was performed on both the co-training method and the self-training method which uses both ChIP-seq and sequence features, and the results can be seen in Figure 5.4. We see that the self-training method is overfitting regardless of the number of iterations being performed, and the co-training method initially overfits as well, but this overfitting decreases with each iteration of the algorithm.

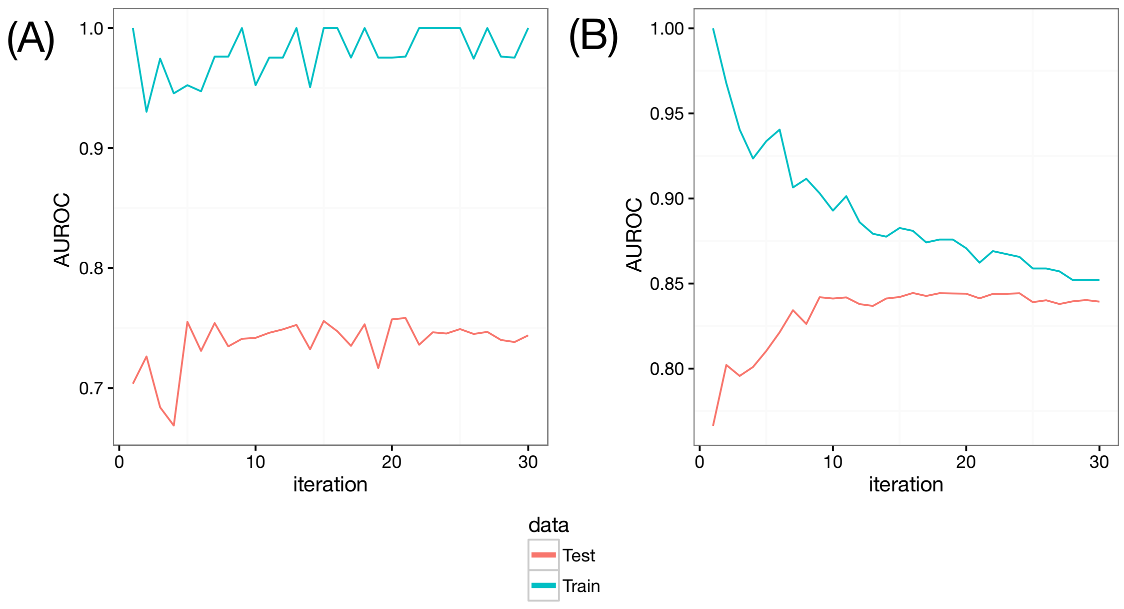


Figure 5.4: Comparison of overfitting with co-training and self-training. To evaluate whether the methods are overfitting, I compared the predictive performance on the training set and test set of (A) self-training with all features and (B) co-training

5.4 Discussion

The enhancer prediction problem is usually addressed by supervised learning algorithms based on chromatin features and/or sequence properties that need a large quantity of labeled examples to perform well. These approaches rely on genome-wide experimental data to define enhancers, making the labeled training set highly dependent on the accuracy of the given technique, as well as specific characteristics of the experiment or high-throughput technology that is used.

We show that the agreement between these experimental methods is very low, which casts doubt on how suitable the results of any single one of these experiments are for use as a gold standard enhancer set. To avoid this problem, we instead require that our gold standard enhancers are identified by three separate methods. The challenge that this stringent approach presents is that we start with a very small set of high confidence enhancers, which makes it difficult to find predictive patterns in those enhancers because we have so few examples to learn from.

We addressed this challenge using co-training, a semi-supervised learning method which uses multiple sets of input features as well as both labeled and unlabeled data. We show that co-training performs the task of predicting enhancer regions using a small training set better than supervised learning as well as self-training, which is another semi-supervised learning method. Additionally, we show evidence that the reason for co-training's better performance is that it reduces overfitting, a major problem when trying to apply machine learning methods to small datasets such as the one we have here. While this wasn't our initial motivation for using co-training, we later found that co-training is known to be more resilient to overfitting than self-training, as mentioned in Aggarwal (2015).

In the future, there are several ways that we could expand upon this analysis. We plan to incorporate other feature sets containing higher information content than dinucleotide frequencies, for example transcription factor binding site motif match scores, in order to further improve the classifier performance as well as biological interpretability. Additionally, it is important to see how this method performs in other cell types in order to confirm that the results observed here generalize to other cell lines, tissues and organisms. We intentionally did not compare our results to other methods that rely on a much larger training set to do their predictions, but it would be useful to compare our results with unsupervised learning methods such as ChromHMM, EpiCSeg or Segway (Ernst and Kellis, 2012; Mammana and Chung, 2015; Hoffman et al., 2012).

Lastly, we should consider different methods for defining what we consider to be high confidence enhancers, especially as more experimental data becomes available or new types of experiments are devised. The Vista enhancer set we used is very small compared to the other enhancer sets, and was the main reason we had such a small training set. Newer experimental methods such as STARR-seq have the goal of providing similar information as the classic enhancer assay, but at a much larger scale, and it would be interesting to use that data in place of the Vista data. Also, our use of the intersection of three experimental methods to define a high confidence set does not take into account how confidently each individual experimental method predicts that region as an enhancer. We could instead incorporate the weights or ranks from each method into an aggregate score for each potential enhancer. Alternatively, we could use a set of regions that were not identified using large scale experimental methods at all, but were rather individually and extensively studied with smaller scale experiments (such as enhancers in the β-globin locus).

Chapter 6

Conclusions

Gene regulation is one of the core biological mechanisms involved in defining cellular identify, and the misregulation of gene expression has been implicated in a wide variety of diseases. This has made the study of the genome, as well as how the genes it encodes are regulated, one of the major topics in biology. Rapid advances in technology have allowed us to more easily read an organism's genome sequence, and additionally observe many other aspects of the cell's state including epigenetic marks such as DNA methylation and chemically modified histones, the three-dimensional structure of chromatin, and active RNA transcription (see Chapter 2). Nevertheless, this abundance of experimental data does not immediately lead to new biological insights. In Chapter 3, we described the field of machine learning and outlined machine learning methods that are well suited to the task of exploring relationships between different types of data, particularly while incorporating the genome's sequence as a predictive feature.

In Chapter 4, we presented a study of non-methylated islands (NMIs) in the genome using newly available experimental data and supervised learning models. We showed that models trained on experimentally determined non-methylated regions can accurately predict the methylation status of a given genomic region based on its DNA sequence, in several species and tissues. These models allowed us to identify differences in the predictive DNA features in cold- versus warm-blooded vertebrates. Additionally, we showed that the tissue-specific methylation can be predicted with high accuracy from DNA sequence as well, and that in colon and placenta methylation is particularly difficult to predict in comparison to other tissues.

Chapter 5 presented the problem of identifying regulatory regions known as enhancers. We highlighted the lack of agreement between several experimental methods that are commonly used to identify putative enhancer regions. Therefore, in contrast to other enhancer prediction methods we chose not to rely on any single experiment to define a set of true enhancer regions, and instead used only those regions that are supported by several experimental methods. This stringent requirement left us with a very small set of true enhancers, so we used a semi-supervised learning approach so that our model could incorporate information from both the known enhancer regions and the rest of the genome. Our predictions are based on two separate input feature sets: ChIP-seq features, which are commonly used to predict enhancers, and DNA sequence information as a second feature set. We show that when using a small input set of known enhancer regions, our method achieves the highest predictive performance when compared to simpler semi-supervised models as well as supervised learning methods.

Overall, we were able to use predictive models to better predict and understand genomic methylation across several species and tissues, as well as to better predict genomic enhancers when starting with very few known examples. These models incorporated DNA sequence data as predictive features, and enabled us to show a correlation between DNA sequence and genomic regions that can influence gene expression. Nevertheless, there are some opportunities for future work.

First, despite the predictive performance of our models it was still difficult to associate the model's performance with specific biological entities. For example, in spite of our efforts to associate predictive k-mer sequences with transcription factors through their binding motifs, it was not possible to make such an association for any of the cold-blooded vertebrate species. In the case of enhancer prediction, our goal was to include cell type-independent signals (histone modifications) as well as cell type-specific signals (DNA sequence features) into the model. However, the optimal predictive features for the model that we built were very simple, just a small set of ChIP-seq experiments and dinucleotides as sequence features. These very simple sequence features made it impossible to speculate on the role of sequence transcription factors in tissue specific enhancer activity using this model, or any other more biologically interesting hypothesis.

In terms of machine learning methods, we could also use methods other than SVMs and logistic regression. Very recently we have seen the rapid development of easy to use libraries for performing deep learning, and deep learning approaches appear to be achieving state-of-the-art performance for most classification tasks where a large amount of data is available. Already, these methods have been applied to the problem of predicting genomic methylation with promising results (Angermueller et al., 2017), and it would be interesting to see how a similar model would perform on the more diverse set of species that were analyzed in this thesis. Additionally, there are deep learning models that can perform semi-supervised learning, but there have been few, if any, applications of these methods to biological problems. Based on the results presented in Chapter 5, it might be worthwhile to apply deep semi-supervised learning to the task of enhancer prediction, while incorporating more complex sequence features.

Bibliography

Aggarwal, C. C. *Data Mining: The Textbook.* 2015, p. 746. ISBN: 9783319141411. DOI: 10.1007/978-3-319-14142-8. arXiv: arXiv:1011.1669v3.

Amano, T. et al. "Chromosomal dynamics at the Shh locus: limb bud-specific differential regulation of competence and active transcription." In: *Developmental cell* 16.1 (Jan. 2009), pp. 47–57. ISSN: 1878-1551. DOI: 10.1016/j.devcel.2008.11.011.

Andersson, R. et al. "An atlas of active enhancers across human cell types and tissues". In: *Nature* 507.7493 (2014), pp. 455–461.

Angermueller, C. et al. "DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning". In: *Genome Biology* 18.1 (2017), p. 67. ISSN: 1474-760X. DOI: 10.1186/s13059-017-1189-z.

Antequera, F. and a. Bird. "Number of CpG islands and genes in human and mouse." In: *Proceedings of the National Academy of Sciences of the United States of America* 90.24 (1993), pp. 11995–11999. ISSN: 0027-8424. DOI: 10.1073/pnas.90.24.11995.

Arnold, C. D. et al. "Genome-Wide Quantitative Enhancer Activity Maps Identified by STARR-seq." In: *Science (New York, N.Y.)* 42.3 (Jan. 2013), pp. 255–9. ISSN: 1095-9203. DOI: 10.1126/science.1232542.

Asai, K., S. Hayamizu, and K. Handa. "Prediction of protein secondary structure by the hidden Markov model". In: *Comput Appl Biosci* 9.2 (1993), pp. 141–146.

Bailey, T. L. "DREME: Motif discovery in transcription factor ChIP-seq data". In: *Bioinformatics* 27.12 (2011), pp. 1653–1659. ISSN: 13674803. DOI: 10.1093/bioinformatics/btr261. arXiv: PMC3106199.

Bailey, T. L. et al. "MEME: Discovering and analyzing DNA and protein sequence motifs". In: *Nucleic Acids Research* 34.WEB. SERV. ISS. (2006), pp. 369–373. ISSN: 03051048. DOI: 10.1093/nar/gkl198.

Barski, A. et al. "High-Resolution Profiling of Histone Methylations in the Human Genome". In: *Cell* 129.4 (2007), pp. 823–837. ISSN: 00928674. DOI: 10.1016/j.cell.2007.05.009. arXiv: NIHMS150003.

Benjamini, Y. and Y. Hochberg. “Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing”. In: *J R Statist Soc B* 57.1 (1995), pp. 289–300.

Bird, A. P. “DNA methylation and the frequency of CpG in animal DNA.” In: *Nucleic acids research* 8.7 (Apr. 1980), pp. 1499–504. ISSN: 0305-1048.

Bird, A. P. and A. P. Wolffe. “Methylation-induced repression–belts, braces, and chromatin.” In: *Cell* 99.5 (Nov. 1999), pp. 451–4. ISSN: 0092-8674.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 0387310738.

Blackledge, N. P. et al. “Bio-CAP: a versatile and highly sensitive technique to purify and characterise regions of non-methylated DNA.” In: *Nucleic acids research* 40.4 (Feb. 2012), e32. ISSN: 1362-4962. DOI: 10.1093/nar/gkr1207.

Blum, A. and T. Mitchell. “Combining Labeled and Unlabeled Data with Co-Training”. In: *Proceedings of the 11th Annual Conference on Computational Learning Theory*. 1998, pp. 92–100.

Bussemaker, H. J., H. Li, and E. D. Siggia. “Regulatory element detection using correlation with expression. TL - 27”. In: *Nature genetics* 27 VN - r.2 (2001), pp. 167–171. ISSN: 1061-4036. DOI: 10.1038/84792.

Chepelev, I. et al. “Characterization of genome-wide enhancer-promoter interactions reveals co-expression of interacting genes and modes of higher order chromatin organization.” In: *Cell research* 22.3 (Mar. 2012), pp. 490–503. ISSN: 1748-7838. DOI: 10.1038/cr.2012.15.

Churchill, G. A. “Hidden Markov Chains and the Analysis of Genome Structure”. In: 16.2 (1992).

Cokus, S. J. et al. “Shotgun bisulphite sequencing of the Arabidopsis genome reveals DNA methylation patterning”. In: *Nature* 452.7184 (2008), pp. 215–219. ISSN: 0028-0836. DOI: 10.1038/nature06745.

Collins, M. and Y. Singer. “Unsupervised Models for Named Entity Classification”. In: *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. 1999, pp. 100–110.

Conlon, E. M. et al. “Integrating regulatory motif discovery and genome-wide expression analysis”. In: *Proceedings of the National Academy of Sciences* 100.6 (2003), pp. 3339–3344. ISSN: 0027-8424. DOI: 10.1073/pnas.0630591100.

Cooper, D. N., M. H. Taggart, and A. P. Bird. “Unmethylated domains in vertebrate DNA.” In: *Nucleic acids research* 11.3 (Feb. 1983), pp. 647–58. ISSN: 0305-1048. DOI: 10.1093/nar/11.3.647.

Cortes, C. and V. Vapnik. “Support-vector networks”. In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 0885-6125. DOI: 10.1007/BF00994018.

Coulondre, C. et al. “Molecular basis of base substitution hotspots in Escherichia coli.” In: *Nature* 274.5673 (Aug. 1978), pp. 775–80. ISSN: 0028-0836.

Cross, S. et al. “Non-methylated islands in fish genomes are GC-poor.” In: *Nucleic acids research* 19.7 (1991), pp. 1469–74. ISSN: 0305-1048.

Das, D., N. Banerjee, and M. Q. Zhang. “Interacting models of cooperative gene regulation”. In: *Proceedings of the National Academy of Sciences* 101.46 (2004), pp. 16234–16239. ISSN: 0027-8424. DOI: 10.1073/pnas.0407365101.

Davuluri, R. V., I. Grosse, and M. Q. Zhang. “Computational identification of promoters and first exons in the human genome.” In: *Nature genetics* 29.4 (2001), pp. 412–7. ISSN: 1061-4036. DOI: 10.1038/ng780.

Deaton, A. M. and A. Bird. “CpG islands and the regulation of transcription.” In: *Genes & development* 25.10 (May 2011), pp. 1010–22. ISSN: 1549-5477. DOI: 10.1101/gad.2037511.

Degroeve, S. et al. “Feature subset selection for splice site prediction”. In: *Bioinformatics* 18 Suppl 2 (2002), S75–S83. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/18.suppl_2.S75.

Derrien, T. et al. “Fast Computation and Applications of Genome Mappability”. In: *PLoS ONE* 7.1 (2012), e30377. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0030377.

Durbin, R. et al. “Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids”. In: *Analysis* (1998), p. 356. ISSN: 0263-6484. DOI: 10.1017/CBO9780511790492. arXiv: 0304372 [math.PR].

Eddy, S. R. “Multiple alignment using hidden Markov models.” In: *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ;*

ISMB. International Conference on Intelligent Systems for Molecular Biology 3 (1995), pp. 114–120. ISSN: 1553-0833. DOI: 7584426.

Elango, N. and S. V. Yi. "DNA Methylation and Structural and Functional Bimodality of Vertebrate Promoters". In: *Molecular Biology and Evolution* 25.8 (Apr. 2008), pp. 1602–1608. ISSN: 0737-4038. DOI: 10.1093/molbev/msn110.

Ernst, J. and M. Kellis. "ChromHMM: automating chromatin-state discovery and characterization." In: *Nature methods* 9.3 (Mar. 2012), pp. 215–6. ISSN: 1548-7105. DOI: 10.1038/nmeth.1906.

Erwin, G. D. et al. "Integrating diverse datasets improves developmental enhancer prediction". In: 37232.615 (Sept. 2013), p. 33. arXiv: 1309.7382.

Friedman, J. H. "Multivariate Adaptive Regression Splines". In: *The Annals of Statistics* 19.1 (1991), pp. 1–67. ISSN: 0090-5364. DOI: 10.1214/aos/1176347963. arXiv: arXiv:1306.3979v1.

Gao, F., B. C. Foat, and H. J. Bussemaker. "Defining transcriptional networks through integrative modeling of mRNA expression and transcription factor binding data. TL - 5". In: *BMC bioinformatics* 5 VN - re.1 (2004), p. 31. ISSN: 1471-2105. DOI: 10.1186/1471-2105-5-31.

Gardiner-Garden, M. and M. Frommer. "CpG islands in vertebrate genomes." In: *Journal of molecular biology* 196.2 (July 1987), pp. 261–82. ISSN: 0022-2836.

Gupta, S. et al. "Quantifying similarity between motifs". In: *Genome Biology* 8.2 (2007), R24. ISSN: 14656906. DOI: 10.1186/gb-2007-8-2-r24.

Harrell, F. E. *Regression Modeling Strategies*. Springer Series in Statistics. New York, NY: Springer New York, 2001. ISBN: 978-1-4419-2918-1. DOI: 10.1007/978-1-4757-3462-1. arXiv: arXiv:1011.1669v3.

Hastie, T., R. Tibshirani, and J. Friedman. "The Elements of Statistical Learning". In: *Elements* 1 (2009), pp. 337–387. ISSN: 03436993. DOI: 10.1007/b94608. arXiv: 1010.3003.

Helgadottir, A. et al. "A Common Variant on Chromosome 9p21 Affects the Risk of Myocardial Infarction". In: *Science* 316.5830 (2007), pp. 1491–1493. ISSN: 0036-8075. DOI: 10.1126/science.1142842. eprint: http://science.sciencemag.org/content/316/5830/1491.full.pdf.

Hinton, G. et al. “Deep Neural Networks for Acoustic Modeling in Speech Recognition”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 82–97. ISSN: 1053-5888. DOI: `10.1109/MSP.2012.2205597`. arXiv: `1207.0580`.

Hocking, R. R. “The Analysis and Selection of Variables in Linear Regression”. In: *Biometrics* 32.1 (1976), p. 1. ISSN: 0006341X. DOI: `10.2307/2529336`. arXiv: `arXiv:1011.1669v3`.

Hoffman, M. M. et al. “Unsupervised pattern discovery in human chromatin structure through genomic segmentation”. In: *Nature methods* 9.5 (2012), pp. 473–476.

Huska, M. R. et al. “Predicting enhancers using a small subset of high confidence examples and co-training”. In: *PeerJ Preprints* 4 (Sept. 2016), e2407v1. ISSN: 2167-9843. DOI: `10.7287/peerj.preprints.2407v1`.

Huska, M. and M. Vingron. “Improved Prediction of Non-methylated Islands in Vertebrates Highlights Different Characteristic Sequence Patterns.” In: *PLOS Computational Biology* 12.12 (Dec. 2016). Ed. by I. Ioshikhes, e1005249. ISSN: 1553-7358. DOI: `10.1371/journal.pcbi.1005249`.

Irizarry, R. a. et al. “The human colon cancer methylome shows similar hypo- and hypermethylation at conserved tissue-specific CpG island shores.” In: *Nature genetics* 41.2 (Feb. 2009), pp. 178–86. ISSN: 1546-1718. DOI: `10.1038/ng.298`.

Jaakkola, T., M. Diekhans, and D. Haussler. “Using the Fisher kernel method to detect remote protein homologies.” eng. In: *Proceedings. International Conference on Intelligent Systems for Molecular Biology* (1999), pp. 149–158. ISSN: 1553-0833 (Print).

Jia, C. and W. He. “EnhancerPred: a predictor for discovering enhancers based on the combination and selection of multiple features Cangzhi Jia & Wenying He”. In: *Nature Publishing Group* 1 (2016), pp. 1–7. ISSN: 2045-2322. DOI: `10.1038/srep38741`.

Johnson, D. S. et al. “Genome-wide mapping of in vivo protein-DNA interactions.” In: *Science (New York, N.Y.)* 316.5830 (2007), pp. 1497–502. ISSN: 1095-9203. DOI: `10.1126/science.1141319`. arXiv: `20`.

Juan, D. et al. “Epigenomic Co-localization and Co-evolution Reveal a Key Role for 5hmC as a Communication Hub in the Chromatin Network of ESCs”. In: *Cell reports* 14.5 (2016), pp. 1246–1257.

Karlić, R. et al. "Histone modification levels are predictive for gene expression." In: *Proceedings of the National Academy of Sciences of the United States of America* 107.7 (Feb. 2010), pp. 2926–31. ISSN: 1091-6490. DOI: `10.1073/pnas.0909344107`.

Keleş, S., M. J. van der Laan, and C. Vulpe. "Regulatory motif finding by logic regression". In: *Bioinformatics* 20.16 (2004), pp. 2799–2811. ISSN: 13674803. DOI: `10.1093/bioinformatics/bth333`.

Kent, W. J. et al. "The Human Genome Browser at UCSC". In: *Genome Research* 12.6 (May 2002), pp. 996–1006. ISSN: 1088-9051. DOI: `10.1101/gr.229102`.

Kim, S. G. et al. "EP-DNN: A Deep Neural Network-Based Global Enhancer Prediction Algorithm". In: *Scientific Reports* 6.1 (2016), p. 38433. ISSN: 2045-2322. DOI: `10.1038/srep38433`.

Kim, T.-K. et al. "Widespread transcription at neuronal activity-regulated enhancers." In: *Nature* 465.7295 (May 2010), pp. 182–7. ISSN: 1476-4687. DOI: `10.1038/nature09033`.

Kodzius, R. et al. "CAGE: cap analysis of gene expression." In: *Nature methods* 3.3 (Mar. 2006), pp. 211–222. ISSN: 1548-7091. DOI: `10.1038/nmeth0306-211`.

Krakau, S., H. Richard, and A. Marsico. "PureCLIP: capturing target-specific protein-RNA interaction footprints from single-nucleotide CLIP-seq data". In: *bioRxiv* (2017). ISSN: 1474-760X. DOI: `10.1101/146704`.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances In Neural Information Processing Systems* (2012), pp. 1–9. ISSN: 10495258. DOI: `http://dx.doi.org/10.1016/j.protcy.2014.09.007`. arXiv: `1102.0183`.

Kuhn, M. "Building predictive models in R using the caret package". In: *Journal of Statistical Software* 28.5 (2008).

Kuksa, P., P.-H. Huang, and V. Pavlovic. "A fast, large-scale learning method for protein sequence classification". In: *8th Int. Workshop on Data Mining in Bioinformatics* (2008), pp. 29–37.

Kundu, D. and G. Murali. "Model selection in linear regression". In: *Computational Statistics and Data Analysis* 22.5 (1996). ISSN: 01679473. DOI: `10.1016/0167-9473(96)00008-4`.

Lawrence, M., S. Daujat, and R. Schneider. "Lateral Thinking: How Histone Modifications Regulate Gene Expression". In: *Trends in Genetics* 32.1 (2016), pp. 42–56. ISSN: 13624555. DOI: `10.1016/j.tig.2015.10.007`.

Lee, D., R. Karchin, and M. A. Beer. "Discriminative prediction of mammalian enhancers from DNA sequence." In: *Genome research* 21.12 (Dec. 2011), pp. 2167–80. ISSN: 1549-5469. DOI: `10.1101/gr.121905.111`.

Lee, T. I. and R. A. Young. *Transcriptional regulation and its misregulation in disease.* 2013. DOI: `10.1016/j.cell.2013.02.014`.

Leslie, C. S. et al. "Mismatch string kernels for discriminative protein classification". In: *Bioinformatics* 20.4 (Mar. 2004), pp. 467–476. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btg431`.

Leslie, C., E. Eskin, and W. S. Noble. "The spectrum kernel: a string kernel for SVM protein classification." In: *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* (Jan. 2002), pp. 564–75. ISSN: 2335-6936.

Leung, M. K. et al. "Deep learning of the tissue-regulated splicing code". In: *Bioinformatics* 30.12 (2014), pp. 121–129. ISSN: 14602059. DOI: `10.1093/bioinformatics/btu277`.

Lewin, H. a. et al. "Every genome sequence needs a good map". In: *Genome Research* 19.11 (2009), pp. 1925–1928. ISSN: 10889051. DOI: `10.1101/gr.094557.109`.

Li, G. et al. "Extensive promoter-centered chromatin interactions provide a topological basis for transcription regulation". In: *Cell* 148.1 (2012), pp. 84–98.

Li, H. and R. Durbin. "Fast and accurate short read alignment with Burrows-Wheeler transform". In: *Bioinformatics* 25 (2009), pp. 1754–1760.

Li, W., D. Notani, and M. G. Rosenfeld. "Enhancers as non-coding RNA transcription units: recent insights and future perspectives". In: *Nature Reviews Genetics* 17.4 (2016), pp. 207–223. ISSN: 1471-0056. DOI: `10.1038/nrg.2016.4`.

Li, Y., C. Y. Chen, and W. W. Wasserman. "Deep feature selection: Theory and application to identify enhancers and promoters". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* Vol. 9029. 2015, pp. 205–217. ISBN: 9783319167053. DOI: `10.1007/978-3-319-16706-0_20`. arXiv: `arXiv:1011.1669v3`.

Lister, R. et al. “Highly Integrated Single-Base Resolution Maps of the Epigenome in Arabidopsis”. In: *Cell* 133.3 (2008), pp. 523–536. ISSN: 00928674. DOI: `10.1016/j.cell.2008.03.029`.

Liu, F. et al. “PEDLA: predicting enhancers with a deep learning-based algorithmic framework”. In: *Scientific Reports* 6.1 (Sept. 2016), p. 28517. ISSN: 2045-2322. DOI: `10.1038/srep28517`.

Logan, B. et al. “Summary for Policymakers”. In: *Climate Change 2013 - The Physical Science Basis*. Ed. by Intergovernmental Panel on Climate Change. Vol. CRL 2001/0. June. Cambridge: Cambridge University Press, 2001, pp. 1–30. ISBN: 9788578110796. DOI: `10.1017/CBO9781107415324.004`. arXiv: `arXiv:1011.1669v3`.

Long, H. K. et al. “Epigenetic conservation at gene regulatory elements revealed by non-methylated DNA profiling in seven vertebrates.” In: *eLife* 2 (Jan. 2013), e00348. ISSN: 2050-084X. DOI: `10.7554/eLife.00348`.

Long, H. K. et al. “Protection of CpG islands from DNA methylation is DNA-encoded and evolutionarily conserved”. In: *Nucleic Acids Research* (2016), gkw258. ISSN: 0305-1048. DOI: `10.1093/nar/gkw258`.

Love, M. I. et al. “Modeling read counts for CNV detection in exome sequencing data”. In: *Statistical Applications in Genetics and Molecular Biology* 10.1 (2011). ISSN: 15446115. DOI: `10.2202/1544-6115.1732`.

Lukashin, A. “GeneMark.hmm: new solutions for gene finding”. In: *Nucleic Acids Research* 26.4 (Feb. 1998), pp. 1107–1115. ISSN: 13624962. DOI: `10.1093/nar/26.4.1107`.

Lupiáñez, D. G. et al. “Disruptions of topological chromatin domains cause pathogenic rewiring of gene-enhancer interactions”. In: *Cell* 161.5 (2015), pp. 1012–1025.

Machanick, P. and T. L. Bailey. “MEME-ChIP: Motif analysis of large DNA datasets”. In: *Bioinformatics* 27.12 (2011), pp. 1696–1697. ISSN: 13674803. DOI: `10.1093/bioinformatics/btr189`. arXiv: `PMC3106185`.

Magnusson, M. et al. “Dynamic Enhancer Methylation - A Previously Unrecognized Switch for Tissue-Type Plasminogen Activator Expression”. In: *Plos One* 10.10 (2015), e0141805. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0141805`.

Mammana, A. and H.-R. Chung. “Chromatin segmentation based on a probabilistic model for read counts explains a large portion of the epigenome”. In: *Genome Biology* 16.1 (Dec. 2015), p. 151. ISSN: 1465-6906. DOI: `10.1186/s13059-015-0708-z`.

Mammana, A. and J. Helmuth. “bamsignals: Extract read count signals from bam files”. In: (2014). R package version 1.2.0.

Martens, J. “Generating Text with Recurrent Neural Networks”. In: *Neural Networks* 131.1 (2011), pp. 1017–1024. ISSN: 1. DOI: `2`. arXiv: `9809069v1 [arXiv:gr-qc]`.

Meissner, A. et al. “Reduced representation bisulfite sequencing for comparative high-resolution DNA methylation analysis”. In: *Nucleic Acids Research* 33.18 (2005), pp. 5868–5877. ISSN: 03051048. DOI: `10.1093/nar/gki901`.

Mendizabal, I. and S. V. Yi. “Whole-genome bisulfite sequencing maps from multiple human tissues reveal novel CpG islands associated with tissue-specific regulation.” In: *Human molecular genetics* 25.1 (2015), pp. 69–82. ISSN: 1460-2083. DOI: `10.1093/hmg/ddv449`.

Müeller-Storm, H. P., J. M. Sogo, and W. Schaffner. “An enhancer stimulates transcription in trans when attached to the promoter via a protein bridge.” In: *Cell* 58.4 (Aug. 1989), pp. 767–77. ISSN: 0092-8674.

Nigam, K. and R. Ghani. “Understanding the Behavior of Co-training”. In: *Proceedings of KDD-2000 Workshop on Text Mining.* 2000.

Nobrega, M. A. “Scanning Human Gene Deserts for Long-Range Enhancers”. In: *Science* 302.5644 (2003), pp. 413–413. ISSN: 0036-8075. DOI: `10.1126/science.1088328`.

Palme, J., S. Hochreiter, and U. Bodenhofer. “KeBABS: An R package for kernel-based analysis of biological sequences”. In: *Bioinformatics* 31.15 (2015), pp. 2574–2576. ISSN: 14602059. DOI: `10.1093/bioinformatics/btv176`.

Pavlidis, P. et al. “Promoter region-based classification of genes.” In: *Pac Symp Biocomput* (2001), pp. 151–163. ISSN: 2335-6936.

Pennacchio, L. A. and E. M. Rubin. “Genomic strategies to identify mammalian regulatory sequences.” In: *Nature reviews. Genetics* 2.2 (Feb. 2001), pp. 100–9. ISSN: 1471-0056. DOI: `10.1038/35052548`.

Pennacchio, L. A. et al. “In vivo enhancer analysis of human conserved non-coding sequences”. In: *Nature* 444.7118 (Nov. 2006), pp. 499–502. ISSN: 0028-0836. DOI: 10.1038/nature05295.

Rajagopal, N. et al. “RFECS: A Random-Forest Based Algorithm for Enhancer Identification from Chromatin State”. In: *PLoS Computational Biology* 9.3 (Mar. 2013). Ed. by M. Singh, e1002968. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1002968.

Rao, S. S. et al. “A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping”. In: *Cell* 159.7 (2014), pp. 1665–1680.

Riloff, E. and R. Jones. “Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping”. In: *Proceedings of the Sixteenth National Conference on Artificial Intelligence*. 1999, pp. 474–479.

Robertson, G. et al. “Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing.” In: *Nature methods* 4.8 (2007), pp. 651–7. ISSN: 1548-7091. DOI: 10.1038/nmeth1068.

Roqueiro, D. et al. “In silico phenotyping via co-training for improved phenotype prediction from genotype.” In: *Bioinformatics* 31.12 (2015), pp. i303–i310.

Sahlén, P. et al. “Genome-wide mapping of promoter-anchored interactions with close to single-enhancer resolution”. In: *Genome Biology* 16.1 (2015), p. 156. ISSN: 1474-760X. DOI: 10.1186/s13059-015-0727-9.

Saito, T. and M. Rehmsmeier. “The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets”. In: *Plos One* 10.3 (2015), e0118432. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0118432.

Salton, G. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Computer Science Series. Addison-Wesley, 1989. ISBN: 9780201122275.

Sandelin, A. et al. “JASPAR: an open-access database for eukaryotic transcription factor binding profiles”. In: *Nucleic Acids Research* 32.90001 (2004), pp. 91D–94. ISSN: 1362-4962. DOI: 10.1093/nar/gkh012.

Saxonov, S., P. Berg, and D. L. Brutlag. “A genome-wide analysis of CpG dinucleotides in the human genome distinguishes two distinct classes of promoters.” In: *Proceedings of the National Academy of Sciences of the United States of*

America 103.5 (Jan. 2006), pp. 1412–7. ISSN: 0027-8424. DOI: `10.1073/pnas.0510310103`.

Schölkopf, B., K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. Computational Molecular Biology. Cambridge, MA, USA: MIT Press, Aug. 2004, p. 410.

Sharif, J. et al. "Divergence of CpG island promoters: A consequence or cause of evolution?" In: *Development, Growth & Differentiation* 52.6 (2010), pp. 545–554. ISSN: 00121592. DOI: `10.1111/j.1440-169X.2010.01193.x`.

Song, F. et al. "Association of tissue-specific differentially methylated regions (TDMs) with differential gene expression." In: *Proceedings of the National Academy of Sciences of the United States of America* 102.9 (2005), pp. 3336–41. ISSN: 0027-8424. DOI: `10.1073/pnas.0408436102`.

Sonnenburg, S. et al. "The SHOGUN Machine Learning Toolbox". In: *Journal of Machine Learning Research* 11 (2010), pp. 1799–1802.

Sutskever, I., O. Vinyals, and Q. Le. "Sequence to sequence learning with neural networks". In: *Advances in Neural Information . . .* (2014), pp. 1–9. ISSN: 09205691. DOI: `10.1007/s10107-014-0839-0`. arXiv: `1409.3215`.

Tibshirani, R. *Regression Selection and Shrinkage via the Lasso*. 1996. DOI: `10.2307/2346178`. arXiv: `11/73273 [1369-7412]`.

Visel, A., J. Bristow, and L. A. Pennacchio. "Enhancer identification through comparative genomics". In: *Seminars in Cell & Developmental Biology* 18.1 (Feb. 2007), pp. 140–152. ISSN: 10849521. DOI: `10.1016/j.semcdb.2006.12.014`.

Visel, A. et al. "Ultraconservation identifies a small subset of extremely constrained developmental enhancers." In: *Nature genetics* 40.2 (Feb. 2008), pp. 158–60. ISSN: 1546-1718. DOI: `10.1038/ng.2007.55`.

Visel, A. et al. "VISTA Enhancer Browser–a database of tissue-specific human enhancers." In: *Nucleic acids research* 35.Database issue (Jan. 2007), pp. D88–92. ISSN: 1362-4962. DOI: `10.1093/nar/gkl822`.

Vishwanathan, S. V. N. and A. J. Smola. "Fast kernels for string and tree matching". In: *Advances in Neural Information Processing Systems 15* (2003), pp. 569–576. DOI: `10.1.1.4.9887`.

Weedon, M. N. et al. “Recessive mutations in a distal PTF1A enhancer cause isolated pancreatic agenesis.” In: *Nature genetics* 46.1 (2014), pp. 61–4. ISSN: 1546-1718. DOI: `10.1038/ng.2826`.

Wikimedia Commons. *File:Central dogma of molecular biology.svg — Wikimedia Commons, the free media repository.* [Online; accessed 13-August-2017]. 2016. URL: `https://commons.wikimedia.org/w/index.php?title=File:Central_dogma_of_molecular_biology.svg`.

— *File:Wiki Bisulfite sequencing Figure 1 small.png — Wikimedia Commons, the free media repository.* [Online; accessed 27-August-2017]. 2016. URL: `https://en.wikipedia.org/w/index.php?title=File:Wiki_Bisulfite_sequencing_Figure_1_small.png&oldid=483285425`.

Wu, H. et al. “Redefining CpG islands using hidden Markov models.” In: *Biostatistics (Oxford, England)* 11.3 (July 2010), pp. 499–514. ISSN: 1468-4357. DOI: `10.1093/biostatistics/kxq005`.

Yarowsky, D. “Unsupervised word sense disambiguation rivaling supervised methods.” In: *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. 1995, pp. 189–196.

Yin, Y. et al. “Impact of cytosine methylation on DNA binding specificities of human transcription factors”. In: *Science* 356.6337 (2017). ISSN: 10959203. DOI: `10.1126/science.aaj2239`.

Young, R. S. et al. “Bidirectional transcription marks accessible chromatin and is not specific to enhancers”. In: (2016). DOI: `10.1101/048629`.

Zeilinger, S. et al. “Tobacco smoking leads to extensive genome-wide changes in DNA methylation.” In: *PloS one* 8.5 (2013), e63812. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0063812`.

Zhang, Y. et al. “Model-based Analysis of ChIP-Seq (MACS)”. In: *Genome Biology* 9.9 (2008), R137. ISSN: 1465-6906. DOI: `10.1186/gb-2008-9-9-r137`.

Zhou, J. and O. G. Troyanskaya. “Predicting effects of noncoding variants with deep learning–based sequence model”. In: *Nature Methods* 12.10 (Aug. 2015), pp. 931–934. ISSN: 1548-7091. DOI: `10.1038/nmeth.3547`. arXiv: `15334406`.

Ziller, M. J. et al. “Charting a dynamic DNA methylation landscape of the human genome.” In: *Nature* 500.7463 (2013), pp. 477–81. ISSN: 1476-4687. DOI: `10.1038/nature12433`. arXiv: `arXiv:1011.1669v3`.

List of Figures

List of Tables

Appendices

Appendix A

Supplementary tables and figures

Table A.1: Comparison of highest weighted 6-mers features from a trained string kernel SVM with known JASPAR core vertebrate motifs using the tool Tomtom. 6-mers were ranked by average feature weight across 5 random splits of the genome into parameter tuning, training and test sets. The top 20 k-mers with the highest weight (contributing most to classifying a sequence as a NMI), and bottom 20 k-mers (contributing most to classifying the sequence as a non-NMI) were used. Only motif matches with a q-value less than 0.1 were kept. The Query ID consists of the position of the k-mer in the feature weight ranking, the k-mer sequence itself, whether it contributes positively or negatively to classifying the sequence as an NMI, and the average feature weight.

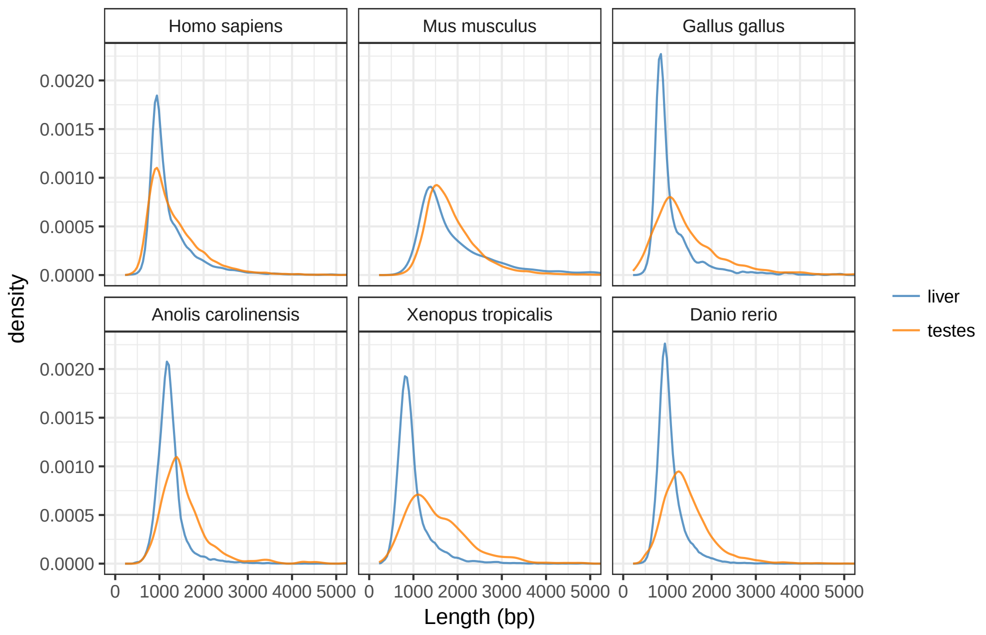
Org	Class	Query ID	Target ID	Offset	p-value	E-value	q-value	Overlap	Query	Target	Strand	GeneSymbol
ac	NMI	5_CCCCCC_pos13.98	MA0753.1	1	0.00	0.01	0.02	6	CCCCCC	CCCCCCCCAC	+	ZNF740
ac	NMI	5_CCCCCC_pos13.98	MA0736.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCGCGAAG	+	GLIS2
ac	NMI	5_CCCCCC_pos13.98	MA0737.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCACGAAG	+	GLIS3
ac	NMI	5_CCCCCC_pos13.98	MA0735.1	3	0.00	0.03	0.02	6	CCCCCC	AGACCCCCCACGAAGC	+	GLIS1
ac	NMI	5_CCCCCC_pos13.98	MA0697.1	2	0.00	0.15	0.06	6	CCCCCC	GACCCCCCGCTGCGC	+	ZIC3
ac	NMI	5_CCCCCC_pos13.98	MA0751.1	2	0.00	0.18	0.06	6	CCCCCC	GACCCCCCGCTGTGC	+	ZIC4
ac	NMI	5_CCCCCC_pos13.98	MA0696.1	2	0.00	0.22	0.06	6	CCCCCC	GACCCCCCGCTGTG	+	ZIC1
ac	NMI	10_GGGGGG_pos12.53	MA0753.1	2	0.00	0.01	0.02	6	GGGGGG	GTGGGGGGGG	-	ZNF740
ac	NMI	10_GGGGGG_pos12.53	MA0736.1	6	0.00	0.03	0.02	6	GGGGGG	CTTCGCGGGGGGTC	-	GLIS2
ac	NMI	10_GGGGGG_pos12.53	MA0737.1	6	0.00	0.03	0.02	6	GGGGGG	CTTCGTGGGGGGTC	-	GLIS3
ac	NMI	10_GGGGGG_pos12.53	MA0735.1	7	0.00	0.03	0.02	6	GGGGGG	GCTTCGTGGGGGGTCT	-	GLIS1
ac	NMI	10_GGGGGG_pos12.53	MA0697.1	7	0.00	0.15	0.06	6	GGGGGG	GCGCAGCGGGGGGTC	-	ZIC3
ac	NMI	10_GGGGGG_pos12.53	MA0751.1	7	0.00	0.18	0.06	6	GGGGGG	GCACAGCGGGGGGTC	-	ZIC4
ac	NMI	10_GGGGGG_pos12.53	MA0696.1	6	0.00	0.22	0.06	6	GGGGGG	CACAGCGGGGGGTC	-	ZIC1
hs	NMI	2_CCGCCC_pos10.31	MA0470.1	5	0.00	0.09	0.09	6	CCGCCC	CCTTCCCGCCC	-	E2F4
hs	NMI	2_CCGCCC_pos10.31	MA0162.2	3	0.00	0.17	0.09	6	CCGCCC	CCCCCGCCCCCGCC	+	EGR1
hs	NMI	2_CCGCCC_pos10.31	MA0079.3	3	0.00	0.19	0.09	6	CCGCCC	GCCCCGCCCCC	+	SP1
hs	NMI	2_CCGCCC_pos10.31	MA0471.1	5	0.00	0.19	0.09	6	CCGCCC	CCTTCCCGCCC	-	E2F6
hs	NMI	2_CCGCCC_pos10.31	MA0516.1	3	0.00	0.25	0.10	6	CCGCCC	GCCCCGCCCCCTCCC	+	SP2
hs	NMI	5_CCCGCC_pos9.74	MA0865.1	3	0.00	0.02	0.03	6	CCCGCC	TTTCCCGCCAAA	+	E2F8
hs	NMI	5_CCCGCC_pos9.74	MA0758.1	4	0.00	0.03	0.03	6	CCCGCC	TTTTCCCGCCAAAA	+	E2F7
hs	NMI	5_CCCGCC_pos9.74	MA0470.1	4	0.00	0.07	0.04	6	CCCGCC	CCTTCCCGCCC	-	E2F4
hs	NMI	5_CCCGCC_pos9.74	MA0471.1	4	0.00	0.07	0.04	6	CCCGCC	CCTTCCCGCCC	-	E2F6
hs	NMI	5_CCCGCC_pos9.74	MA0516.1	2	0.00	0.09	0.04	6	CCCGCC	GCCCCGCCCCCTCCC	+	SP2
hs	NMI	5_CCCGCC_pos9.74	MA0079.3	2	0.00	0.19	0.06	6	CCCGCC	GCCCCGCCCCC	+	SP1
hs	NMI	5_CCCGCC_pos9.74	MA0162.2	2	0.00	0.22	0.06	6	CCCGCC	CCCCCGCCCCCGCC	+	EGR1
hs	NMI	7_GGCGGG_pos9.43	MA0865.1	3	0.00	0.02	0.03	6	GGCGGG	TTTGGCGGGAAA	-	E2F8
hs	NMI	7_GGCGGG_pos9.43	MA0758.1	4	0.00	0.03	0.03	6	GGCGGG	TTTTGGCGGGAAAA	-	E2F7
hs	NMI	7_GGCGGG_pos9.43	MA0470.1	1	0.00	0.07	0.04	6	GGCGGG	GGGCGGGAAGG	+	E2F4
hs	NMI	7_GGCGGG_pos9.43	MA0471.1	1	0.00	0.07	0.04	6	GGCGGG	GGGCGGGAAGG	+	E2F6
hs	NMI	7_GGCGGG_pos9.43	MA0516.1	7	0.00	0.09	0.04	6	GGCGGG	GGGAGGGGGCGGGGC	-	SP2
hs	NMI	7_GGCGGG_pos9.43	MA0079.3	3	0.00	0.19	0.06	6	GGCGGG	GGGGGCGGGGC	-	SP1
hs	NMI	7_GGCGGG_pos9.43	MA0162.2	6	0.00	0.22	0.06	6	GGCGGG	GGCGGGGGCGGGGG	-	EGR1
hs	NMI	8_GGGCGG_pos9.26	MA0470.1	0	0.00	0.09	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F4

Org	Class	Query ID	Target ID	Offset	p-value	E-value	q-value	Overlap	Query	Target	Strand	GeneSymbol
hs	NMI	8_GGGCGG_pos9.26	MA0162.2	5	0.00	0.17	0.09	6	GGGCGG	GGCGGGGGCGGGGG	-	EGR1
hs	NMI	8_GGGCGG_pos9.26	MA0079.3	2	0.00	0.19	0.09	6	GGGCGG	GGGGGCGGGGC	-	SP1
hs	NMI	8_GGGCGG_pos9.26	MA0471.1	0	0.00	0.19	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F6
hs	NMI	8_GGGCGG_pos9.26	MA0516.1	6	0.00	0.25	0.10	6	GGGCGG	GGGAGGGGGCGGGGC	-	SP2
mm	NMI	1_GGGCGG_pos10.96	MA0470.1	0	0.00	0.09	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F4
mm	NMI	1_GGGCGG_pos10.96	MA0162.2	5	0.00	0.17	0.09	6	GGGCGG	GGCGGGGGCGGGGG	-	EGR1
mm	NMI	1_GGGCGG_pos10.96	MA0079.3	2	0.00	0.19	0.09	6	GGGCGG	GGGGGCGGGGC	-	SP1
mm	NMI	1_GGGCGG_pos10.96	MA0471.1	0	0.00	0.19	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F6
mm	NMI	1_GGGCGG_pos10.96	MA0516.1	6	0.00	0.25	0.10	6	GGGCGG	GGGAGGGGGCGGGGC	-	SP2
mm	NMI	2_CCGCCC_pos10.74	MA0470.1	5	0.00	0.09	0.09	6	CCGCCC	CCTTCCCGCCC	-	E2F4
mm	NMI	2_CCGCCC_pos10.74	MA0162.2	3	0.00	0.17	0.09	6	CCGCCC	CCCCCGCCCCCGCC	+	EGR1
mm	NMI	2_CCGCCC_pos10.74	MA0079.3	3	0.00	0.19	0.09	6	CCGCCC	GCCCCGCCCCC	+	SP1
mm	NMI	2_CCGCCC_pos10.74	MA0471.1	5	0.00	0.19	0.09	6	CCGCCC	CCTTCCCGCCC	-	E2F6
mm	NMI	2_CCGCCC_pos10.74	MA0516.1	3	0.00	0.25	0.10	6	CCGCCC	GCCCCGCCCCCTCCC	+	SP2
mm	NMI	5_CCCGCC_pos9.89	MA0865.1	3	0.00	0.02	0.03	6	CCCGCC	TTTCCCGCCAAA	+	E2F8
mm	NMI	5_CCCGCC_pos9.89	MA0758.1	4	0.00	0.03	0.03	6	CCCGCC	TTTTCCCGCCAAAA	+	E2F7
mm	NMI	5_CCCGCC_pos9.89	MA0470.1	4	0.00	0.07	0.04	6	CCCGCC	CCTTCCCGCCC	-	E2F4
mm	NMI	5_CCCGCC_pos9.89	MA0471.1	4	0.00	0.07	0.04	6	CCCGCC	CCTTCCCGCCC	-	E2F6
mm	NMI	5_CCCGCC_pos9.89	MA0516.1	2	0.00	0.09	0.04	6	CCCGCC	GCCCCGCCCCCTCCC	+	SP2
mm	NMI	5_CCCGCC_pos9.89	MA0079.3	2	0.00	0.19	0.06	6	CCCGCC	GCCCCGCCCCC	+	SP1
mm	NMI	5_CCCGCC_pos9.89	MA0162.2	2	0.00	0.22	0.06	6	CCCGCC	CCCCCGCCCCCGCC	+	EGR1
mm	NMI	10_GGCGGG_pos8.95	MA0865.1	3	0.00	0.02	0.03	6	GGCGGG	TTTGGCGGGAAA	-	E2F8
mm	NMI	10_GGCGGG_pos8.95	MA0758.1	4	0.00	0.03	0.03	6	GGCGGG	TTTTGGCGGGAAAA	-	E2F7
mm	NMI	10_GGCGGG_pos8.95	MA0470.1	1	0.00	0.07	0.04	6	GGCGGG	GGGCGGGAAGG	+	E2F4
mm	NMI	10_GGCGGG_pos8.95	MA0471.1	1	0.00	0.07	0.04	6	GGCGGG	GGGCGGGAAGG	+	E2F6
mm	NMI	10_GGCGGG_pos8.95	MA0516.1	7	0.00	0.09	0.04	6	GGCGGG	GGGAGGGGGCGGGGC	-	SP2
mm	NMI	10_GGCGGG_pos8.95	MA0079.3	3	0.00	0.19	0.06	6	GGCGGG	GGGGGCGGGGC	-	SP1
mm	NMI	10_GGCGGG_pos8.95	MA0162.2	6	0.00	0.22	0.06	6	GGCGGG	GGCGGGGGCGGGGG	-	EGR1
mm	NMI	14_CCCCCC_pos7.05	MA0753.1	1	0.00	0.01	0.02	6	CCCCCC	CCCCCCCCAC	+	ZNF740
mm	NMI	14_CCCCCC_pos7.05	MA0736.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCGCGAAG	+	GLIS2
mm	NMI	14_CCCCCC_pos7.05	MA0737.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCACGAAG	+	GLIS3
mm	NMI	14_CCCCCC_pos7.05	MA0735.1	3	0.00	0.03	0.02	6	CCCCCC	AGACCCCCCACGAAGC	+	GLIS1
mm	NMI	14_CCCCCC_pos7.05	MA0697.1	2	0.00	0.15	0.06	6	CCCCCC	GACCCCCCGCTGCGC	+	ZIC3
mm	NMI	14_CCCCCC_pos7.05	MA0751.1	2	0.00	0.18	0.06	6	CCCCCC	GACCCCCCGCTGTGC	+	ZIC4
mm	NMI	14_CCCCCC_pos7.05	MA0696.1	2	0.00	0.22	0.06	6	CCCCCC	GACCCCCCGCTGTG	+	ZIC1
gg	NMI	1_GGGGGG_pos10.59	MA0753.1	2	0.00	0.01	0.02	6	GGGGGG	GTGGGGGGGG	-	ZNF740
gg	NMI	1_GGGGGG_pos10.59	MA0736.1	6	0.00	0.03	0.02	6	GGGGGG	CTTCGCGGGGGGTC	-	GLIS2
gg	NMI	1_GGGGGG_pos10.59	MA0737.1	6	0.00	0.03	0.02	6	GGGGGG	CTTCGTGGGGGGTC	-	GLIS3
gg	NMI	1_GGGGGG_pos10.59	MA0735.1	7	0.00	0.03	0.02	6	GGGGGG	GCTTCGTGGGGGGTCT	-	GLIS1
gg	NMI	1_GGGGGG_pos10.59	MA0697.1	7	0.00	0.15	0.06	6	GGGGGG	GCGCAGCGGGGGGTC	-	ZIC3
gg	NMI	1_GGGGGG_pos10.59	MA0751.1	7	0.00	0.18	0.06	6	GGGGGG	GCACAGCGGGGGGTC	-	ZIC4
gg	NMI	1_GGGGGG_pos10.59	MA0696.1	6	0.00	0.22	0.06	6	GGGGGG	CACAGCGGGGGGTC	-	ZIC1
gg	NMI	2_CCCCCC_pos9.18	MA0753.1	1	0.00	0.01	0.02	6	CCCCCC	CCCCCCCCAC	+	ZNF740
gg	NMI	2_CCCCCC_pos9.18	MA0736.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCGCGAAG	+	GLIS2
gg	NMI	2_CCCCCC_pos9.18	MA0737.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCACGAAG	+	GLIS3
gg	NMI	2_CCCCCC_pos9.18	MA0735.1	3	0.00	0.03	0.02	6	CCCCCC	AGACCCCCCACGAAGC	+	GLIS1
gg	NMI	2_CCCCCC_pos9.18	MA0697.1	2	0.00	0.15	0.06	6	CCCCCC	GACCCCCCGCTGCGC	+	ZIC3
gg	NMI	2_CCCCCC_pos9.18	MA0751.1	2	0.00	0.18	0.06	6	CCCCCC	GACCCCCCGCTGTGC	+	ZIC4

Org	Class	Query ID	Target ID	Offset	p-value	E-value	q-value	Overlap	Query	Target	Strand	GeneSymbol
gg	NMI	2_CCCCCC__pos9.18	MA0696.1	2	0.00	0.22	0.06	6	CCCCCC	GACCCCCCGCTGTG	+	ZIC1
gg	NMI	18_GGGCGG__pos5.43	MA0470.1	0	0.00	0.09	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F4
gg	NMI	18_GGGCGG__pos5.43	MA0162.2	5	0.00	0.17	0.09	6	GGGCGG	GGCGGGGGCGGGGG	-	EGR1
gg	NMI	18_GGGCGG__pos5.43	MA0079.3	2	0.00	0.19	0.09	6	GGGCGG	GGGGGCGGGGC	-	SP1
gg	NMI	18_GGGCGG__pos5.43	MA0471.1	0	0.00	0.19	0.09	6	GGGCGG	GGGCGGGAAGG	+	E2F6
gg	NMI	18_GGGCGG__pos5.43	MA0516.1	6	0.00	0.25	0.10	6	GGGCGG	GGGAGGGGGCGGGGC	-	SP2
ac	notNMI	2_ATAATA__neg9.95	MA0752.1	8	0.00	0.01	0.01	6	ATAATA	TCCATCCCATAATACTC	+	ZNF410
ac	notNMI	6_TATTAT__neg5.84	MA0752.1	3	0.00	0.01	0.01	6	TATTAT	GAGTATTATGGGATGGA	-	ZNF410
ac	notNMI	15_CAGCTG__neg4.37	MA0048.2	2	0.00	0.03	0.01	6	CAGCTG	CGCAGCTGCG	+	NHLH1
ac	notNMI	15_CAGCTG__neg4.37	MA0691.1	2	0.00	0.03	0.01	6	CAGCTG	AACAGCTGAT	+	TFAP4
ac	notNMI	15_CAGCTG__neg4.37	MA0816.1	2	0.00	0.03	0.01	6	CAGCTG	AGCAGCTGCT	+	Ascl2
ac	notNMI	15_CAGCTG__neg4.37	MA0832.1	4	0.00	0.06	0.01	6	CAGCTG	GCAACAGCTGTTGT	+	Tcf21
ac	notNMI	15_CAGCTG__neg4.37	MA0665.1	2	0.00	0.06	0.01	6	CAGCTG	AACAGCTGTT	+	MSC
ac	notNMI	15_CAGCTG__neg4.37	MA0500.1	2	0.00	0.07	0.01	6	CAGCTG	GACAGCTGCAG	+	Myog
ac	notNMI	15_CAGCTG__neg4.37	MA0521.1	2	0.00	0.07	0.01	6	CAGCTG	AACAGCTGCAG	+	Tcf12
ac	notNMI	15_CAGCTG__neg4.37	MA0796.1	3	0.00	0.09	0.01	6	CAGCTG	TGACAGCTGTCA	+	TGIF1
ac	notNMI	15_CAGCTG__neg4.37	MA0499.1	2	0.00	0.19	0.02	6	CAGCTG	TGCAGCTGTCCCT	+	Myod1
ac	notNMI	15_CAGCTG__neg4.37	MA0667.1	2	0.00	0.23	0.02	6	CAGCTG	AACAGCTGTT	+	MYF6
ac	notNMI	15_CAGCTG__neg4.37	MA0797.1	3	0.00	0.36	0.03	6	CAGCTG	TGACAGCTGTCA	+	TGIF2
ac	notNMI	15_CAGCTG__neg4.37	MA0782.1	3	0.00	0.86	0.07	6	CAGCTG	TGACAGGTGTCA	+	PKNOX1
ac	notNMI	17_TTTTAT__neg3.87	MA0465.1	0	0.00	0.02	0.05	6	TTTTAT	TTTTATGGCTT	-	CDX2
ac	notNMI	20_AACAAT__neg3.73	MA0087.1	1	0.00	0.04	0.05	6	AACAAT	AAACAAT	-	Sox5
ac	notNMI	20_AACAAT__neg3.73	MA0077.1	1	0.00	0.15	0.05	6	AACAAT	GAACAATGG	-	SOX9
ac	notNMI	20_AACAAT__neg3.73	MA0866.1	0	0.00	0.21	0.05	6	AACAAT	AACAATGGTAGTGTT	+	SOX21
ac	notNMI	20_AACAAT__neg3.73	MA0869.1	0	0.00	0.21	0.05	6	AACAAT	AACAATTTCAGTGTT	+	Sox11
ac	notNMI	20_AACAAT__neg3.73	MA0870.1	0	0.00	0.21	0.05	6	AACAAT	AACAATAACATTGTT	+	Sox1
ac	notNMI	20_AACAAT__neg3.73	MA0084.1	3	0.00	0.21	0.05	6	AACAAT	GTAAACAAT	+	SRY
ac	notNMI	20_AACAAT__neg3.73	MA0867.1	1	0.00	0.23	0.05	6	AACAAT	GAACAATTGCAGTGTT	+	SOX4
ac	notNMI	20_AACAAT__neg3.73	MA0868.1	0	0.00	0.23	0.05	6	AACAAT	AACAATGTGCAGTGTT	+	SOX8
dr	notNMI	14_CCGGAA__neg3.66	MA0028.2	1	0.00	0.03	0.01	6	CCGGAA	ACCGGAAGTG	+	ELK1
dr	notNMI	14_CCGGAA__neg3.66	MA0475.2	1	0.00	0.03	0.01	6	CCGGAA	ACCGGAAGTG	+	FLI1
dr	notNMI	14_CCGGAA__neg3.66	MA0759.1	1	0.00	0.03	0.01	6	CCGGAA	ACCGGAAGTA	+	ELK3
dr	notNMI	14_CCGGAA__neg3.66	MA0760.1	1	0.00	0.03	0.01	6	CCGGAA	ACCGGAAGTG	+	ERF
dr	notNMI	14_CCGGAA__neg3.66	MA0762.1	2	0.00	0.04	0.01	6	CCGGAA	AACCGGAAATA	+	ETV2
dr	notNMI	14_CCGGAA__neg3.66	MA0473.2	3	0.00	0.05	0.01	6	CCGGAA	AACCCGGAAGTG	+	ELF1
dr	notNMI	14_CCGGAA__neg3.66	MA0641.1	3	0.00	0.05	0.01	6	CCGGAA	AACCCGGAAGTG	+	ELF4
dr	notNMI	14_CCGGAA__neg3.66	MA0640.1	3	0.00	0.05	0.01	6	CCGGAA	AACCCGGAAGTAA	+	ELF3
dr	notNMI	14_CCGGAA__neg3.66	MA0474.2	1	0.00	0.06	0.01	6	CCGGAA	ACCGGAAGTG	+	ERG
dr	notNMI	14_CCGGAA__neg3.66	MA0062.2	0	0.00	0.07	0.01	6	CCGGAA	CCGGAAGTGGC	+	Gabpa
dr	notNMI	14_CCGGAA__neg3.66	MA0598.2	3	0.00	0.09	0.01	6	CCGGAA	AACCCGGAAGTA	+	EHF
dr	notNMI	14_CCGGAA__neg3.66	MA0156.2	1	0.00	0.09	0.01	6	CCGGAA	ACCGGAAGTG	+	FEV
dr	notNMI	14_CCGGAA__neg3.66	MA0763.1	1	0.00	0.09	0.01	6	CCGGAA	ACCGGAAGTG	+	ETV3
dr	notNMI	14_CCGGAA__neg3.66	MA0076.2	1	0.00	0.11	0.01	6	CCGGAA	GCCGGAAGTGG	-	ELK4
dr	notNMI	14_CCGGAA__neg3.66	MA0098.3	1	0.00	0.12	0.01	6	CCGGAA	ACCGGAAGTG	+	ETS1
dr	notNMI	14_CCGGAA__neg3.66	MA0761.1	1	0.00	0.12	0.01	6	CCGGAA	ACCGGAAGTA	+	ETV1
dr	notNMI	14_CCGGAA__neg3.66	MA0764.1	1	0.00	0.15	0.02	6	CCGGAA	ACCGGAAGTA	+	ETV4
dr	notNMI	14_CCGGAA__neg3.66	MA0765.1	1	0.00	0.35	0.04	6	CCGGAA	ACCGGAAGTG	+	ETV5
dr	notNMI	14_CCGGAA__neg3.66	MA0136.2	2	0.00	0.48	0.05	6	CCGGAA	ACCCGGAAGTA	+	ELF5

Org	Class	Query ID	Target ID	Offset	p-value	E-value	q-value	Overlap	Query	Target	Strand	GeneSymbol
hs	notNMI	10_CACCTG_neg4.62	MA0103.2	3	0.00	0.03	0.01	6	CACCTG	CCTCACCTG	+	ZEB1
hs	notNMI	10_CACCTG_neg4.62	MA0745.1	1	0.00	0.03	0.01	6	CACCTG	ACACCTGTT	-	SNAI2
hs	notNMI	10_CACCTG_neg4.62	MA0522.2	2	0.00	0.03	0.01	6	CACCTG	AACACCTGCT	+	TCF3
hs	notNMI	10_CACCTG_neg4.62	MA0824.1	2	0.00	0.03	0.01	6	CACCTG	TACACCTGTC	+	ID4
hs	notNMI	10_CACCTG_neg4.62	MA0830.1	2	0.00	0.03	0.01	6	CACCTG	CGCACCTGCT	+	TCF4
hs	notNMI	10_CACCTG_neg4.62	MA0743.1	3	0.00	0.06	0.02	6	CACCTG	AACCACCTGTTGCTC	-	SCRT1
hs	notNMI	10_CACCTG_neg4.62	MA0744.1	1	0.00	0.10	0.03	6	CACCTG	CCACCTGTTGCAT	-	SCRT2
hs	notNMI	10_CACCTG_neg4.62	MA0820.1	2	0.00	0.12	0.03	6	CACCTG	ACCACCTGTT	+	FIGLA
hs	notNMI	10_CACCTG_neg4.62	MA0783.1	3	0.00	0.20	0.04	6	CACCTG	TGACACCTGTCA	-	PKNOX2
hs	notNMI	20_CAGGTG_neg4.08	MA0103.2	0	0.00	0.03	0.01	6	CAGGTG	CAGGTGAGG	-	ZEB1
hs	notNMI	20_CAGGTG_neg4.08	MA0745.1	2	0.00	0.03	0.01	6	CAGGTG	AACAGGTGT	+	SNAI2
hs	notNMI	20_CAGGTG_neg4.08	MA0522.2	2	0.00	0.03	0.01	6	CAGGTG	AGCAGGTGTT	-	TCF3
hs	notNMI	20_CAGGTG_neg4.08	MA0824.1	2	0.00	0.03	0.01	6	CAGGTG	GACAGGTGTA	-	ID4
hs	notNMI	20_CAGGTG_neg4.08	MA0830.1	2	0.00	0.03	0.01	6	CAGGTG	AGCAGGTGCG	-	TCF4
hs	notNMI	20_CAGGTG_neg4.08	MA0743.1	6	0.00	0.06	0.02	6	CAGGTG	GAGCAACAGGTGGTT	+	SCRT1
hs	notNMI	20_CAGGTG_neg4.08	MA0744.1	6	0.00	0.10	0.03	6	CAGGTG	ATGCAACAGGTGG	+	SCRT2
hs	notNMI	20_CAGGTG_neg4.08	MA0820.1	2	0.00	0.12	0.03	6	CAGGTG	AACAGGTGGT	-	FIGLA
hs	notNMI	20_CAGGTG_neg4.08	MA0783.1	3	0.00	0.20	0.04	6	CAGGTG	TGACAGGTGTCA	+	PKNOX2
xt	notNMI	3_CCCCCC_neg5.80	MA0753.1	1	0.00	0.01	0.02	6	CCCCCC	CCCCCCCCAC	+	ZNF740
xt	notNMI	3_CCCCCC_neg5.80	MA0736.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCGCGAAG	+	GLIS2
xt	notNMI	3_CCCCCC_neg5.80	MA0737.1	2	0.00	0.03	0.02	6	CCCCCC	GACCCCCCACGAAG	+	GLIS3
xt	notNMI	3_CCCCCC_neg5.80	MA0735.1	3	0.00	0.03	0.02	6	CCCCCC	AGACCCCCCACGAAGC	+	GLIS1
xt	notNMI	3_CCCCCC_neg5.80	MA0697.1	2	0.00	0.15	0.06	6	CCCCCC	GACCCCCCGCTGCGC	+	ZIC3
xt	notNMI	3_CCCCCC_neg5.80	MA0751.1	2	0.00	0.18	0.06	6	CCCCCC	GACCCCCCGCTGTGC	+	ZIC4
xt	notNMI	3_CCCCCC_neg5.80	MA0696.1	2	0.00	0.22	0.06	6	CCCCCC	GACCCCCCGCTGTG	+	ZIC1
xt	notNMI	20_CCCCCA_neg2.96	MA0737.1	3	0.00	0.04	0.03	6	CCCCCA	GACCCCCCACGAAG	+	GLIS3
xt	notNMI	20_CCCCCA_neg2.96	MA0753.1	3	0.00	0.04	0.03	6	CCCCCA	CCCCCCCCAC	+	ZNF740
xt	notNMI	20_CCCCCA_neg2.96	MA0735.1	4	0.00	0.05	0.03	6	CCCCCA	AGACCCCCCACGAAGC	+	GLIS1
mm	notNMI	15_CCATGG_neg4.45	MA0600.2	5	0.00	0.26	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX2
mm	notNMI	15_CCATGG_neg4.45	MA0509.1	4	0.00	0.31	0.08	6	CCATGG	GTTGCCATGGCAAC	+	Rfx1
mm	notNMI	15_CCATGG_neg4.45	MA0138.2	7	0.00	0.38	0.08	6	CCATGG	TTCAGCACCATGGACAGCGCC	+	REST
mm	notNMI	15_CCATGG_neg4.45	MA0510.2	5	0.00	0.38	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX5
mm	notNMI	15_CCATGG_neg4.45	MA0798.1	5	0.00	0.38	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX3
mm	notNMI	15_CCATGG_neg4.45	MA0799.1	5	0.00	0.51	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX4
gg	notNMI	3_CCATGG_neg2.00	MA0600.2	5	0.00	0.26	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX2
gg	notNMI	3_CCATGG_neg2.00	MA0509.1	4	0.00	0.31	0.08	6	CCATGG	GTTGCCATGGCAAC	+	Rfx1
gg	notNMI	3_CCATGG_neg2.00	MA0138.2	7	0.00	0.38	0.08	6	CCATGG	TTCAGCACCATGGACAGCGCC	+	REST
gg	notNMI	3_CCATGG_neg2.00	MA0510.2	5	0.00	0.38	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX5
gg	notNMI	3_CCATGG_neg2.00	MA0798.1	5	0.00	0.38	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX3
gg	notNMI	3_CCATGG_neg2.00	MA0799.1	5	0.00	0.51	0.08	6	CCATGG	CGTTGCCATGGCAACG	+	RFX4

(A)



(B)

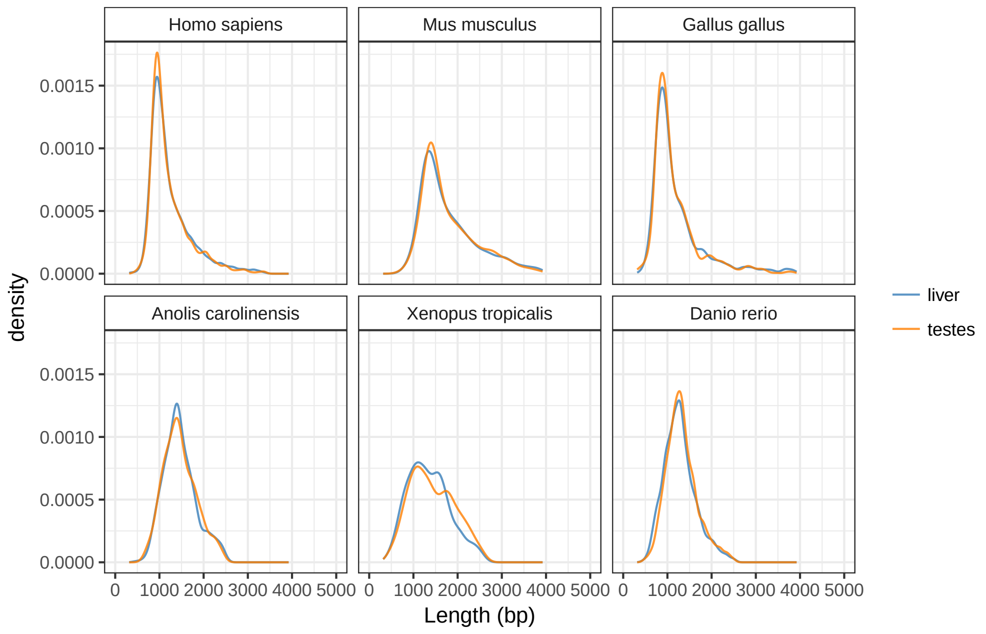
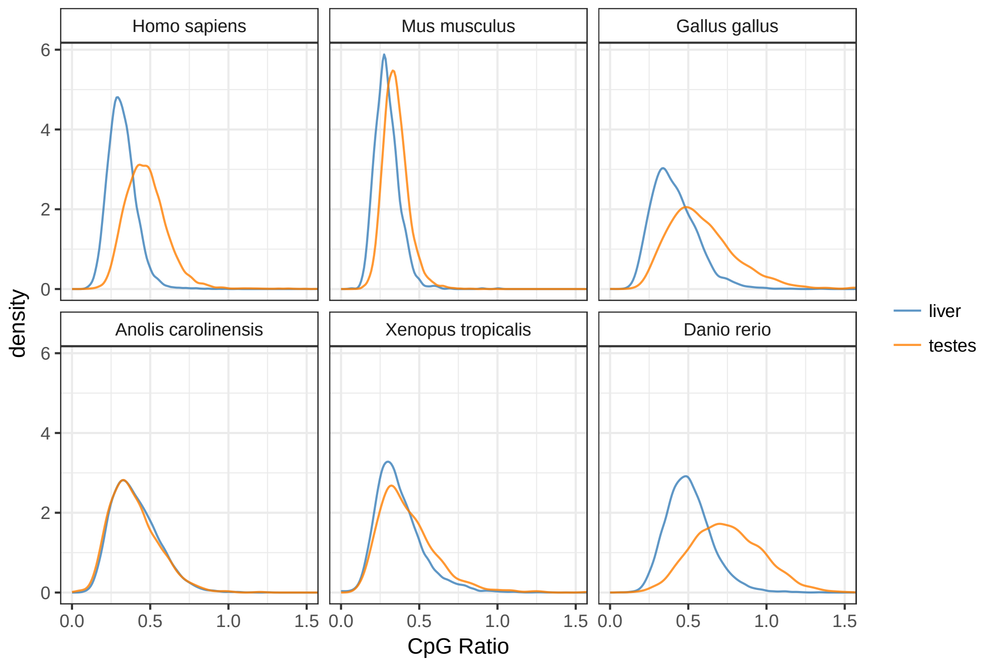


Figure A.1: NMI length distributions in liver and testes. (A) before subsampling (B) after subsampling.

(A)



(B)

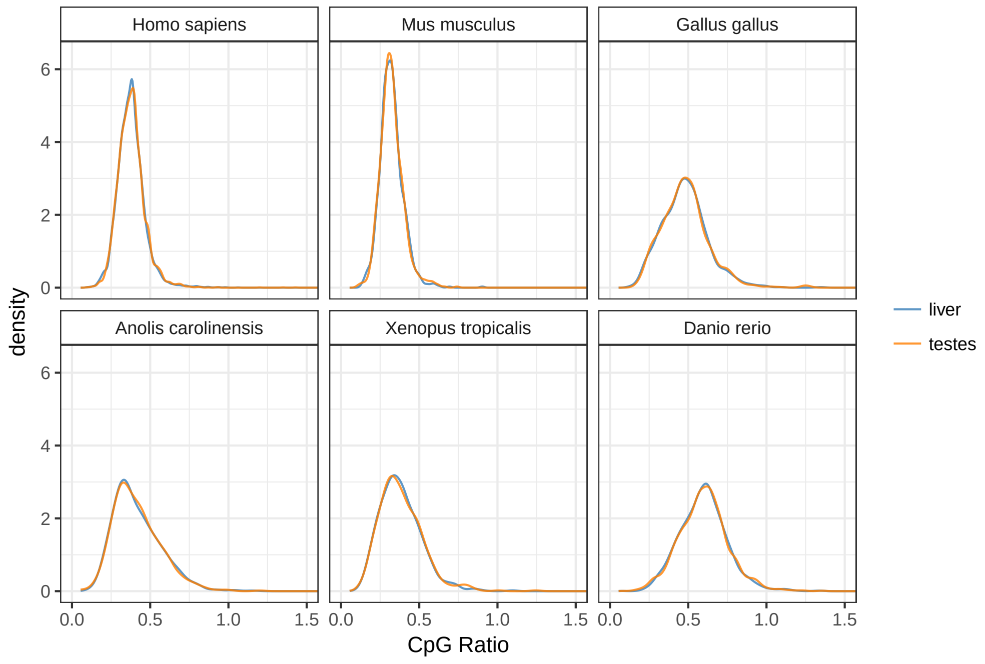


Figure A.2: NMI CpG ratio distributions in liver and testes. (A) before subsampling (B) after subsampling.

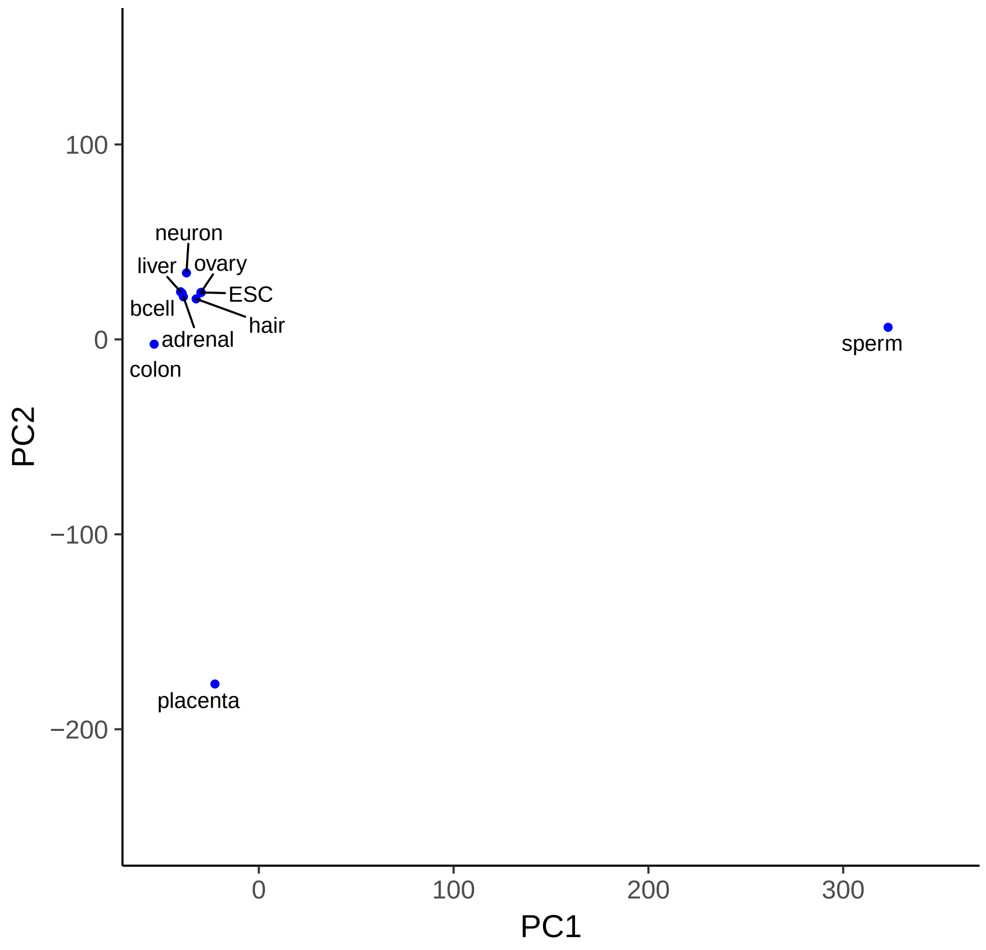


Figure A.3: PCA plot of NMI absence/presence across 10 tissues. Principal components analysis plot of the binary matrix of NMI absence and presence values across ten human tissues. The first two principal components are shown.

Appendix B

Zusammenfassung

In dieser Arbeit untersuchen wir den Einfluss der DNA-Sequenz auf Elemente des Genoms, welche die Genregulation beeinflussen. Zu diesem Zweck nutzen wir Ansätze aus dem Bereich des Maschinellen Lernens, da dieser etablierte Methoden zur Mustererkennung in Datensequenzen bereitstellt, welche wir hier auf Nukleotidsequenzen anwenden.

Zunächst zeigen wir, dass wir die Lage und die Gewebsspezifität von experimentell bestimmten nichtmethylierten Regionen des Genoms mit hoher Genauigkeit vorhersagen können, indem wir die DNA-Sequenz der Region verwenden. Diese Analyse basiert auf neu entwickelten experimentellen Methoden zur Bestimmung von genomweiten DNA-Methylierungen, deren Entwicklung die CpG-Inseln als Grundlage zur Vorhersage von nichtmethylierte Regionen abgelöst hat. Wir demonstrieren die hohe Vorhersageleistung unserer Methode anhand von zwei Geweben in jeweils sechs verschiedenen Vertebratenspezies sowie auch in zehn Humangeweben. Darüberhinaus übertrifft unser Ansatz die Leistung anderer existierender Methoden, welche zur Identifizierung von CpG-Inseln entwickelt wurden.

Des Weiteren präsentieren wir einen neuen Ansatz zur computerbasierten Vorhersage von genomischen Enhancern. Im Gegensatz zu bestehenden Methoden kombinieren wir die Ergebnisse von verschiedenen komplementären experimentellen Methoden um die Menge von Enhancern zu definieren, welche als Mustervorlage des Lernprozesses dient. Außerdem wird ein spezieller Algorithmus des Maschinellen Lernens genutzt, das Co-Training, welches es erlaubt, zum Trainieren des Prädiktors sowohl eine kleine Menge von Enhancerregionen mit hohem Konfidenzniveau als auch den Rest des Genoms zu integieren. Die Vorhersage der Enhancer basiert auf Daten von ChIP-seq Experimenten und der DNA-Sequenz jeder Region. Wir sind in der Lage zu zeigen, dass unser Ansatz eine bessere Vorhersageleistung erreicht als andere Methoden und, dass das Co-Training für diese Art von Problemen besonders gut geeignet ist, da es das Problem der Überanpassung reduziert.

Appendix C

Summary

In this thesis we explore the influence of DNA sequence on genomic elements that are involved in the regulation of genes. We approach this topic using tools from the field of machine learning, which provides established methods for identifying patterns in sequential data, in this case sequences of nucleotides.

First, we show that the location and tissue-specificity of experimentally determined non-methylated regions of the genome can be predicted with high accuracy using the regions' DNA sequence. This analysis relies on new experimental methods that have been used to measure DNA methylation genome-wide, and their development has led to a shift away from relying on CpG islands as a proxy for non-methylated genomic regions. We demonstrate the high predictive performance of our method in two tissues across six different vertebrate species, as well as in ten human tissues, and show that the method we use outperforms existing methods that were designed to identify CpG islands.

Next, we present a new approach to computationally predicting genomic enhancers. In contrast to existing methods, we combine the results of multiple complementary experimental methods to define the set of enhancers from which to learn patterns, and we use a machine learning method called co-training to enable us to incorporate this small set of high confidence enhancer regions as well as the rest of the genome into the training of our predictor. The enhancers are predicted based on both experimental data from ChIP-seq experiments and the DNA sequence of each region. We are able to show that our method achieves better predictive performance than other methods, and that co-training is particularly well suited for this problem because it is able to reduce the problem of overfitting.

Appendix D

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen verwendet habe. Ich erkläre weiterhin, dass ich die vorliegende Arbeit oder deren Inhalt nicht in einem früheren Promotionsverfahren eingereicht habe.

Berlin, den 6.2.2018