ADVANCED MACHINE LEARNING APPROACHES FOR CHARACTERIZATION OF TRANSCRIPTIONAL REGULATORY ELEMENTS AND GENOME-WIDE ASSOCIATIONS

A Dissertation
Presented to
The Academic Faculty

By

Hamid Reza Hassanzadeh

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Department of Interactive Computing

Georgia Institute of Technology

May 2020

ADVANCED MACHINE LEARNING APPROACHES FOR CHARACTERIZATION OF TRANSCRIPTIONAL REGULATORY ELEMENTS AND GENOME-WIDE ASSOCIATIONS

Approved by:

Date Approved: Nov 20, 2019

In Memory of My Grandfather

ACKNOWLEDGMENTS

The work presented in this dissertation would have not been possible to do without the advice, support, and encouragement of many others.

First and foremost, I would like to thank my advisors Dr. Charles Isbell and Dr. Gregory Gibson for supporting me to pursue and finish my thesis work and research. Without their impactful feedbacks and guidances it was impossible for me to pick the right research direction and reach to where I'm today.

Many thanks to my PhD committee, Dr. Konstantinos Dovrolis, Dr. Peng Qiu, and Dr. Denis Tsygankov, for taking the time to review my thesis and provide valuable feedbacks for improving my dissertation.

I would also like to thank my past and current colleagues for the time they spent with me, to help me, to teach me, and to share valuable insights with me, so that I can handle obstacles that I have encountered during my PhD career. A big thank you to Sini Nagpal, Maggie Fisher, Biao Zeng, Po-Yen Wu, Merdad Farajtabar, Gena Tang, and many others whom I might have received any academic or non-academic help during my journey to a PhD degree.

Many thanks to other faculty members and staff whom I am indebted to in one way or another. Dr. King Jordan, Dr. Leigh D. Bottomley, Dr. Ali Adibi, Dr. Hamid Garmestani, Lisa Redding, and many others, for helping me out in several occasions when I ran into different challenges a PhD student may confront. I would also need to thank several Georgia Tech professors and researcher whom I have learnt a lot from, during my past collaborations with them, especially, Dr. Mark Borodovsky, Dr. May Wang, and Dr. Alex Lomsadze.

Last but not least, I would like to thank my family for their help and prayers during all these years, especially, thanks to my wife Ghazal whom undoubtedly I share the credit with, as well as my mom and dad. The last thank you goes to the last member of my family, my adorable daughter, who fully cooperated with daddy to write his dissertation and brought love, and energy to his life!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

BN Batch Normalization. 59

CAGE Consortium for the Architecture of Gene Expression. 73

CNN Convolutional Neural Networks. 15

DAG Directed Acyclic Graph. 75, 76, 79, 100

DNA DeoxyRibonucleic Acid. 1, 2, 3, 4, 5, 11

DPR Dirichlet Process Regression. 77, 103

eQTLs Expression Quantitative Trait Loci. 68, 72

FC Fully Connected. 112

GWA Genome-Wide Association. 11

HMM Hidden Markov Model. 25

LD Linkage Disequilibrium. 13, 138

LSTM Long Short-Term Memory. 17, 33

MAF Minor Allele Frequency. 12

PBM Prortein Binding Microarrays. 5, 8, 9, 26, 61, 133

PWM	Position Weight Matrix. 26
RNA	Ribonucleic Acid. 4
RNN	Recurrent Neural Network. 17
SGD	Stochastic Gradient Descent. 39
SNP	Single Nucleotide Polymorphism. 12
SNV	Single Nucleotide Variant. 12
TF	Transcription Factor. 20
t-SNE	t-Distributed Stochastic Neighbor Embedding. xvi, 128, 130

SUMMARY

The deep learning revolution has initiated a surge of remarkable achievements in diverse research areas where large volumes of data that underlie complex processes exist. Despite the successful application of deep models in solving certain problems in the Biomedical and Bioinformatics domains, the field has not brought any promise in solving many other challenging problems that deal with the genomic complexities. The goal of my Ph.D. research has been to develop advanced machine learning techniques to address two relevant challenging problems in the Bioinformatics domain, namely, the characterization of transcriptional regulatory elements and, modeling genome-wide associations and linkage disequilibrium using genomic and evolutionary annotation of variants.

Genome codes for almost all biological phenomena that take place inside living cells. One such key interactions is the association between transcription factors and a number of degenerate binding sites on DNA which facilitate initiation of transcription of genes. While each protein can potentially bind to any site on the DNA, it is the strength of this binding that plays the key role in the initiation process. Predicting these binding sites as well as binding affinities, are two interesting and yet challenging problems that remain largely unsolved. Yet, we know that the cell machineries constantly identify such sites on DNA with near perfect accuracy. The last two decade witnessed production of multiple in-vivo and in-vitro high-throughput technologies for elucidating these interactions. Protein Binding Microarrays (PBM) have been one of the most effective in-vitro technologies developed so far. The result of PBM experiments, however, are not easily interpretable and require advanced downstream analysis tools to discover the patterns of bindings. In the first half of my thesis, I will develop a series of computational methods that can learn such patterns from data generated by this technology, using tools and techniques from the natural language and image processing domains. I will also show the superiority of my proposed pipelines in predicting binding patterns and affinity.

The second part of my thesis is devoted to developing methods for modeling of genome-wide associations and the linkage disequilibrium. Both of these tasks pose similar challenges that restrict our ability in utilizing recent advances in deep learning research. Specifically, when dealing with GWA studies, we are often bound by high dimensionality of variants data, a significant degree of missing information (i.e. missing heritability), high complexity weak patterns to learn, and relatively small datasets. As a consequence, the state-of-the-art approaches for GWAS that are used in practice are different variations of linear models. In my thesis, I showed that part of the failure in learning higher-capacity models can be attributed to how we are training such models. Specifically, I showed that using Siamese networks and tools from graph theory we can achieve a performance higher or on par with the state-of-the-art Bayesian non-parametric approaches. Being successful in learning weak relationships using the proposed model, I then extended my approach to show that there is a relation between variants annotations and their underlying haplotype structure, which was not known before. Existence of such a relationship can increase the power of GWA models and if proved biologically will have important implications in population genetics.

CHAPTER 1
INTRODUCTION AND BACKGROUND

In this section I provide a concise overview of some of the biological facts and processes as well as computational algorithms and techniques that are necessary to understand the work done in the next chapters. I decided to organize this chapter into three main sections, the first two of which provide a high-level biological background for the work done while the last one introduces the reader to the emerging field of deep learning. Sections 1.1 and 1.2, while tightly connected, merit independent treatments as each relates to the work done in a separate chapter. In Section 1.3, I provide the required background on machine learning in general, and deep learning in particular, as much as it is necessary to understand the approaches introduced in each of the following chapters.

1.1 A Concise Introduction to Molecular Biology

Molecular biology is the science that is concerned with the interactions (at different levels) that take place in the cells of living organisms. Cells are the smallest living entities in all living organisms that we know of today. The majority of all complicated biological phenomena take place in cells. Cells are composed of different compartments that are each in charge of producing, controlling and regulating numerous complex interactions and interrelations. Despite the complexity of all these phenomena, they are all encoded in the Cell in a remarkably large message that is called the DeoxyRibonucleic Acid (DNA) which is built by an alphabet (i.e. A, T, C and G nucleotides) of only size four. Notwithstanding the simplicity of the coding's building blocks, the biological processes in cells are often extremely complicated due to the complex relationships between the confounding environmental stimuli, the existing message, and the manner in which it is transcribed or translated into products and many other molecules that exist in the cell.

Prokaryotes and Eukaryotes: Two Major Categories of Living Organisms

Every organism belongs to either of the two main categories, Prokaryotes and Eukaryotes (Fig. 1.1) which diverged from a primitive cell during a long evolutionary process that started 3.5 billion years ago. As opposed to the Prokaryotic cells, Eukaryotic cells possess a nucleus, which hosts the cell's DNA. In Prokaryotic cells, on the other hand, the DNA resides in the cell's cytoplasm. DNA has a double helix structure which gives it the advantage of remaining in a low-energy stable state. Each helix in DNA is composed of three key components, the sugar molecule, the phosphate group and the base. A base can be one of the four nucleotides, adenine (A), thymine (T), cytosine (C), or guanine (G) attached to the next base via its sugar-phosphate backbone. In Prokaryotes, DNA is often contained in one circular chromosome, while in Eukaryotes, DNA is packed into multiple linear chromosomes. Since the studies reported in this thesis concern the genetics and biology of Eukaryotic genomes, the rest of this text concerns only these genomes, unless otherwise stated.

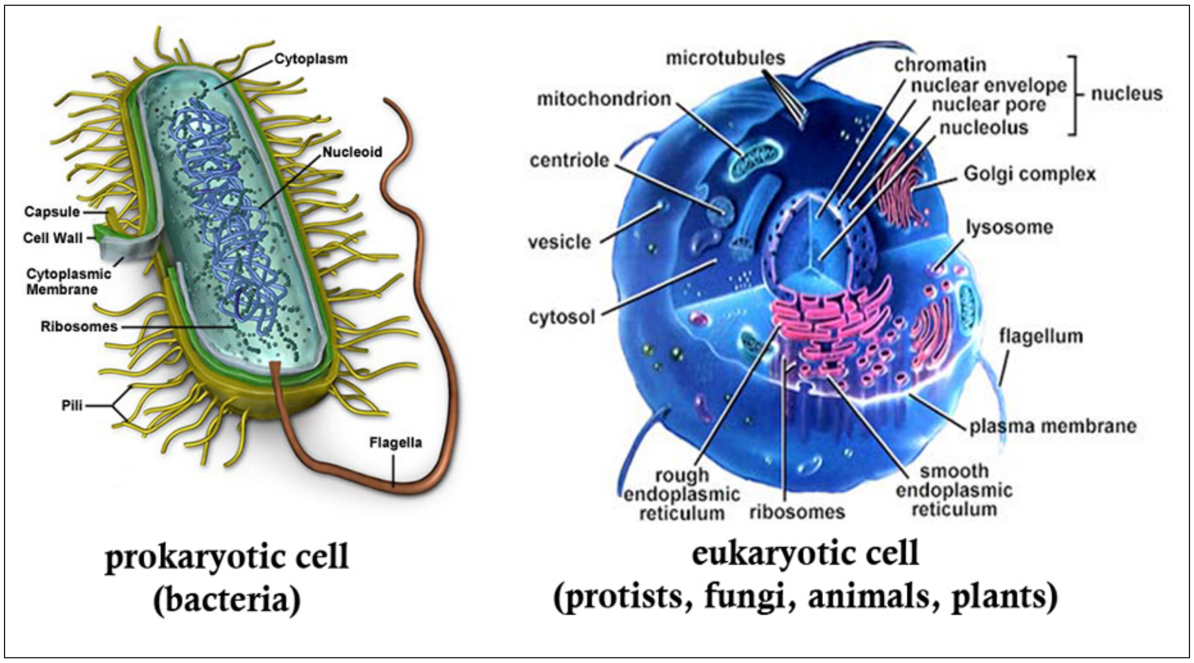


Figure 1.1: Eukaryotic vs Prokaryotic cells.

Image source: https://www.thinglink.com/scene/728336081086316545

DNA and DNA Packaging in Cells

The diploid human genome contains about 6 billion base pairs packed into 46 chromosomes (i.e. 23×2), with each base pair being about 0.34 nanometers long. Since a typical human body has about 50 trillion cells this will amount to over 600 times the distance between the Earth and the Sun [1]. This means that the DNA is hosted in cells in a compact form that leaves yet enough room for other organelles and numerous large molecules.

Chromosomal DNA is packed in multiple levels to form a compact structure called chromatin that fits in the cell nucleus. Chromatin plays key roles in cell functions and maintenance such as the regulation of gene expression and DNA replication. Histone proteins are the key elements that make this compact form possible. Two copies of each of the four positively charged histone proteins (H2A, H2B, H3, and H4) come together to make a histone octamer that the negatively charged DNA double helix wraps around, with a periodicity of approximately 1.7 turns (equal to 146 base pairs). The resulting structure, which is the fundamental unit of chromatin, is called a nucleosome. An auxiliary H1 protein then completes the second turn by wrapping another 20 base pairs around the nucleosome, forming a chromatosome. On average, there are hundreds of thousands of nucleosomes in each chromosome that are connected to each other through linker DNA that is not wrapped around histone proteins. Chromatin is next coiled into a thicker, 30-nm in diameter, fiber which results in a next level of packaging of DNA. This fiber will then fold into loops that are 300 nm long on average and finally, in a next level of compression, these loops fold again to generate a 250 nm-wide fiber, which is again coiled to produce the underlying chromosome.

1.1.1 The Central Dogma of Molecular Biology

The central dogma of molecular biology is the key corner stone of molecular biology that was first stated in 1957 [2] by Francis Crick and remains mostly correct today[1]. It describes the flow of genetic information in biological systems. The central dogma, in its revised form, details the "residue-by-residue transfer of sequential information". According to the dogma, there are three major information-carrying biopolymers, DNA, Ribonucleic Acid (RNA), and protein, establishing nine conceivable possibilities of information transfer. From those, three are not believed to occur, three can occur in rare cases or under special conditions, while the remaining three constantly occur in the cells. Figure 1.2 depicts the general and specific flows. The non-general flow of information includes DNA synthesis from RNA through the reverse transcription process, RNA replication through which the RNA is copied from RNA, and protein synthesis directly from a DNA template (i.e. without the use of an intermediate mRNA -short for messenger RNA). While the aforementioned three transfers have important implications, this dissertation is more concerned about the general information transfers in the cells (also called Watson's dogma), hence, a short summary of the remaining three flows, as related to the subject of next chapters, is given below.

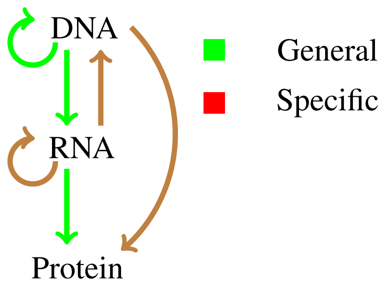


Figure 1.2: Information flow according to the revised central dogma of biology: green and brown arrows show the general and specific transfers, respectively.

[1]The central dogma of biology was later re-stated and refined [3] due to later discoveries such as retroviruses.

1.1.2 From DNA to RNA to Protein

Watson's central dogma of biology states a two-step process for transferring a DNA message into a protein. In the first step, DNA is transcribed into an RNA message called mRNA. This step takes place in the cell nucleus using RNA polymerase. The transcribed mRNAs are then exported outside of the nucleus for the next transfer pathway, translation. Translation is done by ribosomes in the cell's cytoplasm. Each of these steps requires an exactly timed elaborate contribution of different machineries, enzymes, and molecules for a successful delivery of the product at the end of the pathway. As in this dissertation I'm not concerned about the translation process, I will only focus on the transcription part in this text (more details can be found in [4].)

DNA can be conceived as a long script that encodes tens of thousands of recipes (messages) for normal operation of cells. Most of these recipes are what we call genes. Genes usually code for proteins, but some of them code for RNAs that play different roles such as structural or regulatory roles. Despite the small number of nucleotides used in DNA, deciphering the overall effect of these recipes is extremely involved for a number of reasons. Almost all biological pathways are orchestrated via an interplay of multiple gene products (proteins or RNAs) in an elaborate way. Time and the order of compilation is often a key factor in the formation process of many of such pathways. The cell itself is a closed loop system where a change in one input can result in feedback that complicates the cell's behavior further. Moreover, not all genes are expressed at all times and in all cells. In fact, the expression (or level of expression) of many of the genes are cell/tissue dependent and can vary according to the presence of different stimuli. Furthermore, the role of many gene products is not exactly known or is only partially known. In fact, a significant number of genes cooperate in different biological pathways, sometimes playing different roles in each. Last but not least, other chemical elements such as minerals can play a role in advancing a biological pathway. These altogether, make the cell an extremely complicated system to model and calls for a multitude of different high-throughput technologies such as Prortein

Binding Microarrays (PBM) to read cell's status under different conditions and to facilitate making accurate models. Section 1.1.3 overviews a series of available technologies that are used for identifying and quantifying protein-DNA interactions, as related to the work presented in this dissertation.

Transcription of Genes in Eukaryotic Genomes

Transcription is the process by which certain regions of genes on the genome are scanned by an RNA polymerase and a sequence of ribonucleotides that is complementary to the gene under transcription is made. In most Eukaryotes there are three different types of RNA polymerases (some plants have five), which are called Pol I, Pol II, and Pol III. RNA polymerases are made up of multiple subunits which have been well-conserved during evolution. Pol I, and II transcribe specific types of RNA-encoding genes (Pol I transcribes rRNA[2] genes while Pol III transcribes the tRNA genes and a few others). Pol II, on the other hand, is the most studied polymerase and is the one that this thesis is concerned with as it is involved in transcription of most protein coding genes.

Transcription of genes by RNA polymerase II takes place in three phases of initiation, elongation, and termination. In the initiation phase, the RNA polymerase, along with other initiation factors (also known as transcription factors - TFs), land on a DNA region called promoter, which can be as wide as a few kilo bases, making a complex called the pre-initiation complex, which then undergoes some structural changes that allow initiation to start. In vitro, transcription factors are sufficient to initiate transcription as the DNA has no chromatin structure and is well exposed to the polymerase and TFs. In vivo, however, the DNA is wrapped around the nucleosomes and other factors are also required to make transcription possible. As a result of the complex formation, the DNA around the transcription start site denatures and a ”transcription bubble” on the template strand (the strand that serves as the recipe for the transcription) is formed. The RNA polymerase now starts

[2]The large ribosomal RNA which is the RNA component of the ribosome.

synthesizing the first few ribonucleotides of the precursor mRNA (aka pre-mRNA) but in a quite inefficient way. During this phase, several efforts to continue transcription and proceed to the elongation phase may fail and lead to release of the short synthesized transcript and restart of transcription. By the time the enzyme synthesizes a sequence of length longer than around 10 base pairs it has managed to escape the promoter and proceed to the next phase, the elongation phase. Escaping the promoter involves two steps, ADP hydrolysis (to generate the required energy), and phosphorylation of the polymerase[3]. Phosphorylation of the Pol II tail helps it to leave the transcription factor behind as it escapes the promoter and start the elongation phase, in which the precursor mRNA is synthesized, and subsequently moved to the termination phase in which the synthesized pre-mRNA and the polymerase are released. Each of these two phases require participation of other factors and triggering of other events which, though important, are not of central interest for the purposes of my research.

As explained, transcription factors are key regulators of the transcription process and from the computational biology perspective have important implications. For example, a key step in building gene regulatory networks is to find the regulatory role of TFs, which starts by finding the specific locations where they may bind. Different TFs bind to different binding sites (e.g. enhancers, promoters, etc.) of certain target genes in a relatively specific but yet degenerate way. This makes characterizing the corresponding binding sites and site motifs a challenge. Not only that, it is often desirable to model the binding specificity (or relative affinity) of TFs on different DNA sites, which is even more challenging. Different models can be used for characterizing these motifs and their specificities (e.g. position weight matrices - PWMs [5], see Chapter 2 for a review).

[3]The largest subunit of Pol II has a carboxy-terminal domain (CTD) often referred to as the polymerase tail. CTD has a series of heptapeptide repeats (Tyr-Ser-Pro-Thr-Ser-Pro-Ser) which varies genome by genome. These repeats have sites for phosphorylation by certain kinases.

1.1.3 In-Vivo and In-Vitro Technologies for Identifying Protein-DNA interactions

Methods for prediction of transcription factor binding sites can be grouped into two main categories, the in-vivo and the in-vitro approaches. *in-vitro* methods are low in throughput, but can provide extra information that can help unravel more complexities. Specifically, in-vivo methods can capture events that happen under specific conditions (e.g. tissue, cell type, time point, etc.) and in a more realistic environment (i.e. a functioning cell). Moreover, the most prominent in-vivo techniques (i.e. ChiP-based assays) can predict the exact location of binding sites[4]. In-vivo techniques are also able to unravel cooperative binding (see section 2.1.1) that impacts both the binding process and its specificity.

On the other side of the spectrum are the in-vitro methods. While in-vitro approaches that are available today are not able to pinpoint binding motifs, many of them are high/mid-throughput techniques and as a result are well suited for large-scale characterization of protein binding sites. PBM and SELEX-based methods are the two most prominent categories of high throughput in-vitro methods for identifying protein binding specificity. Since in-vitro techniques cannot spot the exact location of binding sites of TFs on DNA, they require further downstream analysis for inferring binding motifs pertaining to each experiment. Therefore, advanced computational aid for characterization of binding motifs is an integral part of the high-throughput in-vitro techniques. In the past decade, several statistical and machine learning based approaches have been proposed to address this question. These methods (especially the most recent ones), have promised performance boosts in their prediction performance but have their own limitations, too. In chapter 2 I overview such methods and discuss the shortcomings pertaining to each. I will then propose a new method that addresses some of these limitations and discuss the opportunities for future extensions. Since in the thesis I specifically focus on PBM[5] a brief introduction to this technology is given bellow.

[4]Although this is limited by genomic composition of sequences.

[5]While we apply our proposed approach to PBM data, it should work for other in-vitro data, with minor modifications.

Protein-Binding Microarrays (PBM)

PBM is an in-vitro high-throughput technique that allows rapid characterization of TF binding sites in a scalable way. PBM is based on the microarray technology and utilizes design of so-called de Bruijn[6] sequences of order k (where k is usually 10,) that are split into different probes of length 60nt. Each probe is synthesized as a single stranded-oligonucleotide sequence, in multiple copies, that are attached to a slide via a thymidine linker, followed by 24 nucleotides that are common across all probes. The remainder of each probe contains a 35-base subsequence of the designed de Bruijn sequence. The probe sequence in each spot shares $k-1$ nucleotides with the next neighboring spot, which allows full coverage of all k-mers. These single-strand probes are then extended through a Cy5-labeled universal primer[7] and dNTPs (di-nucleotide triphosphate), during the assay. Moreover, a small proportion of Cy3-conjugated dUTPs are also added to the reaction. This serves two purposes, first, it can be used as a measure of fidelity of the primer extension to exclude bad spots and second, the signal intensity that results from fluorescent illumination of Cy3-dUTPs is used to quantify the quantity of dsDNA in each probe. This intensity is expected to be linearly proportional to the number of adenines in the template, however, is also dependent on its nucleotide composition [6]. Therefore, a linear model is learned from trinucleotides that are present in each probe to find the right ratio for proper normalization of binding signal intensities. Finally, the binding signal intensity is measured through tagging of the TF with glutathione S-transferase (GST) at the N-terminus and subsequent staining with fluorophore-conjugated anti-GST antibody and quantification of the fluorescent signal with laser scanners. Figure 1.3 depicts main steps for designing a PBM experiment.

The de Bruijn sequences are designed using the linear-feedback shift registers method. De Bruijn sequences are not unique and many such sequences can be designed. In fact, multiple arrays with different de Bruijn sequences are often assayed in PBMs to test for

[6]A de Bruijn sequence of order k is a compact form that contains all k-mers exactly once.

[7]The primer is the reverse complement of the first 25nt of the array probes.

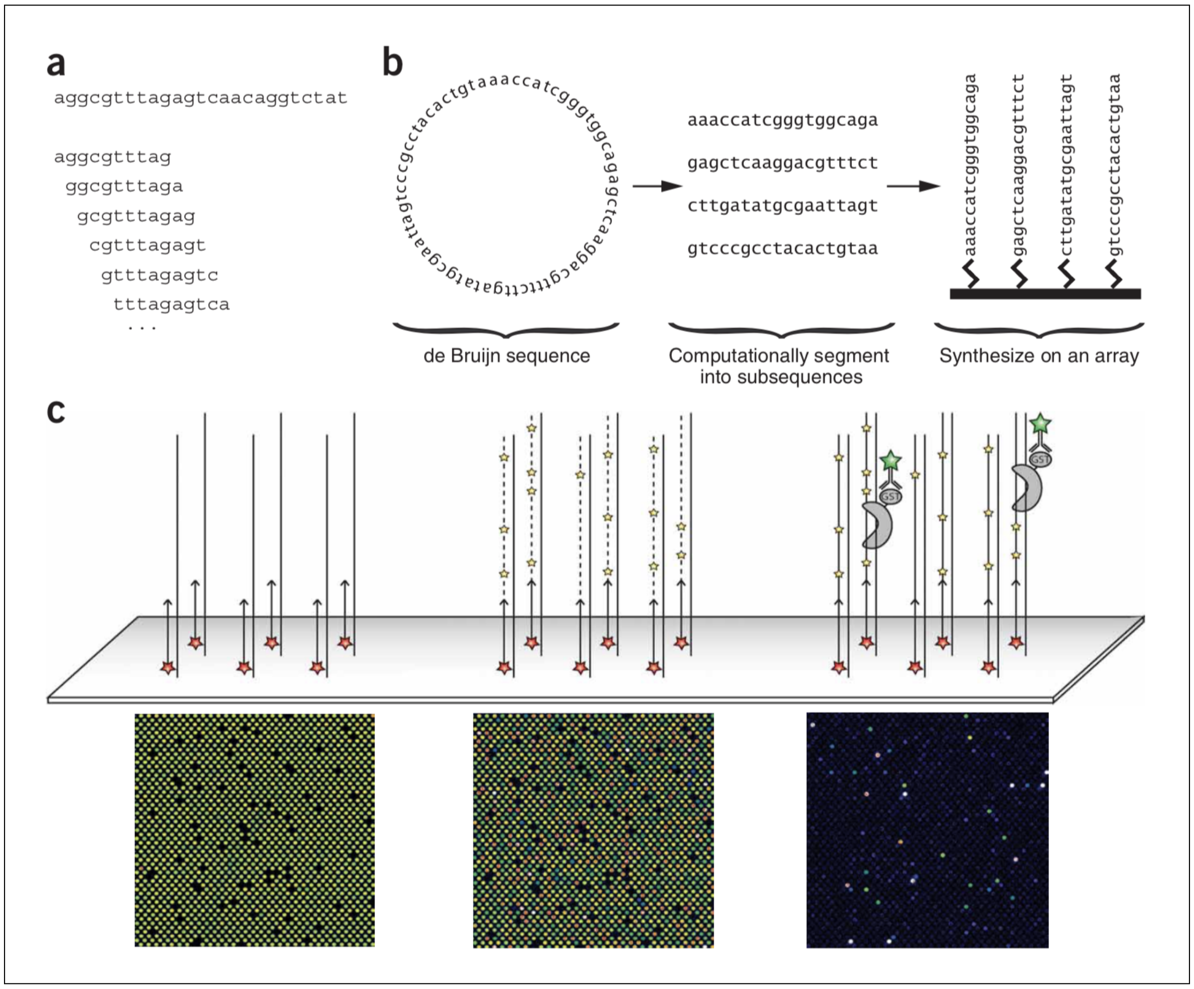


Figure 1.3: Steps to design a universal microarray of protein binding microarray experiments. Source: [6].

replicability of the experiment. A microarray of 44,000 spots can be used to assay the motifs of length 10 nucleotides with all possible single gaps. Note that this is not possible to do if we were to generate one probe for each k-mer[8].

Despite all its unique features, PBM suffers from a few key shortcomings as listed below:

1. The binding signal intensity in each spot is a non-linear function of different confounding factors (e.g. the prep and instrument's parameters) besides the probe's se-

[8]In that case, for a single experiment that covers all ungapped k-mers, we will need $4^{10} > 1,000,000$ probes which is not possible to do with microarrays.

quence context. Therefore, learning a dividing factor with linear regression is sub-optimal.

2. The binding affinity of each k-mer depends also on its position and orientation with respect to the slide surface.

3. Measured specificities are due to the contribution of possibly several k-mers with different degrees of affinity. Consequently, binding signal intensities even without any noise, do not represent the affinity of any individual k-mer.

Notwithstanding these limitations, PBM remains one of the most useful in-vitro, high-throughput technologies for scalable and fast quantification of direct TF-DNA interactions. Therefore, it is crucial to come up with accurate computational techniques to decipher the data generated by PBM experiments, in light of the above-mentioned challenges. In Chapter 2 I design models that address the last two shortcomings listed above, using advanced techniques in machine learning.

1.2 A Concise Introduction to Genome-Wide Association Studies

In genetics, Genome-Wide Association (GWA) study refers to any study that investigates a genome-wide set of genetic variants among different individuals to identify genetic variants that are associated with some trait of interest. Most GWA analyses are limited to bi-allelic variants that influence human traits/diseases, however, the study of any type of variant in any organism may fall under the same name.

1.2.1 Single Nucleotide Variant/Polymorphism

A Single Nucleotide Polymorphism (SNP) is a substitution of a nucleotide at a specific position on the DNA. The substitution can take place on different types of loci, e.g. intergenic, intronic or genic and can have different impacts on the cellular processes which themselves often lead to different phenotypic effects. These effects can range from a specific hair color to chronic disease or risk of such diseases. By convention, a substitution is called Single Nucleotide Polymorphism (SNP) if each of its variations (called alleles) are present in a population in some appreciable degree (e.g. $> 5\%$). These variants must be passed down from ancestors, hence, somatic mutations are not counted as SNPs. On the other hand, all other substitutions (such as somatic mutations) fall under the more general category ofSingle Nucleotide Variant (SNV).

The majority of SNPs are present in two forms only (two alleles), one that is present in a lesser proportion, i.e. the Minor Allele Frequency (MAF) allele, and one that is more common in the population. Since the majority of differences between genomes of individuals from the same organisms are SNPs, it is a widely accepted fact that most of the heritable phenotypic differences among individuals can be explained, or at least partly explained, by differences in the alleles of SNPs that each individual possess. This in fact has resulted in a number of success stories for diagnosis and prognosis or susceptibility prediction of diseases from Cystic Fibrosis [7] to Type 2 Diabetes,. The challenge is that there are millions of SNPs in the genomes of most organisms with extremely intricate relationships between them. Disregarding the uncertainty that is present in this setting due to the existence of other factors such as the environment, randomness, noisy data (e.g. sequencing errors), and etc., these variants are not associated to the phenotypes in a simple univariate way, rather, a large number of them may play a role and in conjunction with each other, in a complicated way. Unraveling the aforementioned complex relationship requires extremely high-capacity models that themselves need a huge amount of data to be trained properly, and which won't available at least in the near future. Even if there were enough data to learn from, non-genetic aspects (generically called ”environment”), which these models can't explain, remain a bottleneck. Therefore, GWA analyses are often limited in scope and are based on relatively simple linear models that can only statistically explain a limited set of observations.

These observations altogether challenge the applicability and feasibility of machine learning approaches to this mostly unknown domain. To the extent that no advanced machine learning model has been able to surpass the performance of available low-capcity Bayesian approaches in this domain. In the second part of my research in particular, I attempt to address some of these shortcomings with the help of techniques that are being utilized in design of deep learning based models in other domains, and show that they can offer promise in this domain, too. Specifically, we will present new solutions to two important problems that we discuss in Chapters 3 and 4 and make quantitative comparisons with the state-of-the-art, when possible.

1.2.2 Linkage Disequilibrium (LD)

In population genetics, Linkage Disequilibrium (LD) is referred to the non-random association of alleles in populations. Under an independent and random distribution of alleles in a haplotype[9], the probability of alleles A, and B, pertaining to two SNPs in two different loci, co-occurring together in a population is expected to be $p_{AB} = p_A p_B$ where p_A and p_B are allele frequencies in the given population for the two SNPs, respectively. In general, this is not however the case for every pair of SNPs, due to factors such as natural selection, mutation, genetic recombination, genetic drift, mating patterns, genetic linkage and population structure.

One way to quantify the level of LD between alleles A and B, is through D_{AB}, the coefficient of linkage disequilibrium, defined as the deviation of the observed from the expected haplotype frequencies (i.e. $D_{AB} = p_{AB} - p_A p_B$). In case this quantity is non-zero (whether positive or negative), we call alleles A and B to be in LD with each other and thus, the probability of each possible haplotype of the two loci will be according to the Table 1.1.

The fact that the range of this coefficient depends on the allele frequencies in the pop-

[9]Haplotype is a set of loci located in one haploid genome.

ulation renders it an inefficient metric for comparisons. The Lewontin coefficient, $D' = D/D_{max}$, can be used instead to normalize the former coefficient by its theoretical maximum, which is $min(p_{Ab}, p_{aB})$ if $D < 0$ and $min(p_{AB}, p_{ab})$, otherwise. Another useful measure is the Pearson correlation coefficient which is defined as $r^2 = D^2/(p_A \times p_B \times p_a \times p_b)$ and ranges between 0 and 1. Variety of evolutionary factors have shaped the LD structure of the genomes that we have today, a survey of which can be found in [8].

Table 1.1: Observed frequency of haplotypes as a function of the Linkage disequilibrium metric and expected allele frequencies.

Haplotype	Observed Frequency
AB	$p_A \times p_B + D$
Ab	$p_A \times (1 - p_B) - D$
aB	$(1 - p_A) \times p_B - D$
ab	$(1 - p_A) \times (1 - p_B) + D$

1.3 An Introduction to Deep Learning

Deep learning (also known as hierarchical learning) is a class of machine learning algorithm that is based on the artificial neural networks. The main idea behind deep learning is that multiple levels of representations can be learned successively from representations of the previous layers, through non-linear transformations [9]. Such networks, almost always, benefit from the gradient descent algorithm to adapt their weights so that each layer learns a more abstract representation from the previous layers. For example, when that task is to classify images, the lower layers of such networks often learn trivial features such as existence of edges, while later layers can inform concepts such as the existence of a specific object in the image. Since the term deep learning was coined, an overwhelming number of non-linear components, network architectures and tricks for training accurate models have been suggested, many of them appear to work remarkably well when dealing with complex data. In this dissertation, I deploy some of them to address problems of interest and we will show how proper use of them leads to improvements in prediction results. Here I give a brief introduction to two important classes of model that are very popular among the deep learning community, namely, the convolutional neural networks and the recurrent neural networks, and postpone introduction of some others, to the chapters where they are used.

1.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN), also known as ConvNets, are often used for image and text data. CNN have been shown to perform surprisingly well on data where features values are locally dependent, which is usually the case for text, image, and some other types of data. CNN is an extension of the regular fully connected layers with the major change that certain neurons share weights with each other and are shifted in time or space. This reduces the number of parameters significantly and allows the model to pick up patterns that recur all over the image, text, etc. Another way to represent a CNN is through a number of kernels that are applied to a local region of the input, followed by a non-linear activation, and then shifted by a pre-specified stride, to generate output feature maps. CNNs often utilize multiple kernels and hence generate feature maps of depth more than one. Often times we use one or more pooling layers to reduce the dimensionality of feature maps. This helps to reduce the model parameters and avoid overfitting [10]. Figure 1.4 shows the architecture of AlexNet, the first successful application of deep learning in the image processing domain. AlexNet [11] had a pivotal role in the introduction of deep learning for researchers and was with a large margin, the winning model for the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC), one of the most popular contests being held annually.

Since the introduction of AlexNet, deeper and deeper models, along with newer components and tricks in learning such models, keep being introduced by researchers as well as big companies such as Google and Microsoft, which has lead to ground-breaking records in the field. Today's CNN-based models often take advantage of one or multiple of techniques such as dropout [12], skip connections [13], inception modules [14], batch normalization [15], and etc. We will also employ some of these in Chapters 2 and 4 where we use some of these techniques to learn patterns of protein-DNA binding and linkage disequilibrium.

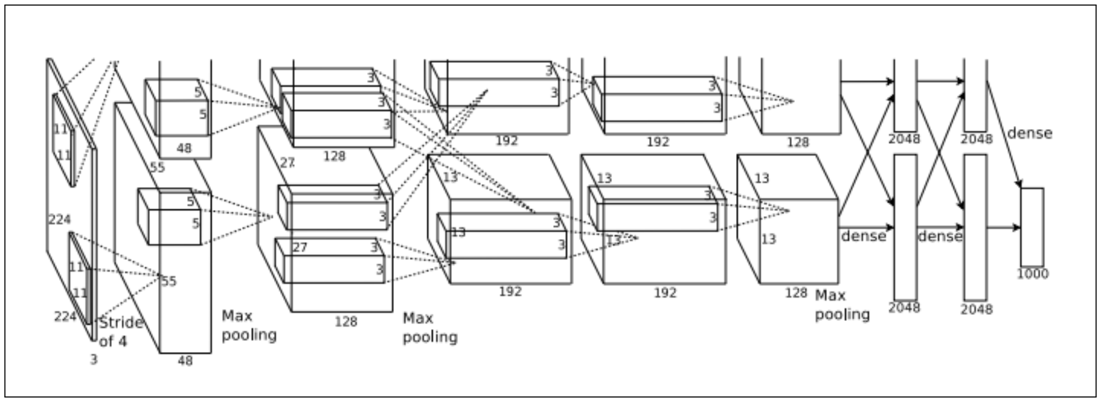


Figure 1.4: AlexNet model architecture. Image from [11].

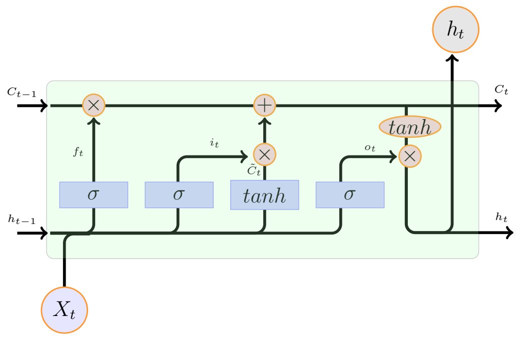
1.3.2 Recurrent Neural Networks and Long Short-Term Memory Networks

Recurrent Neural Network (RNN) models are well known for their ability to learn temporal patterns. These models were introduced even before the spatial convolutional models [16] and have been widely explored in different domains. The principle behind RNNs is that sequential signals which exhibit stationary features in time can be learnt via a recurrent network that unrolls as we proceed through time/position/etc. The unrolled units share the same set of weights as they learn the same task reused at different time points. Despite its mathematical elegance, RNNs in their vanilla form, failed to surpass shallow models due to their limitation in propagating back the error gradients towards distant time points, an issue which is often referred to as the ”vanishing gradient” effect. That is, gradients tend to become prohibitively small as they reach the distant past, as a result of which the long-range temporal dependencies become increasingly hard to capture. In 1997, Hochreiter et. al. [17] proposed a variation of RNNs, the Long Short-Term Memory (LSTM) networks, that could efficiently solve the vanishing gradient problem by introducing a gating mechanism in a well-behaved and differentiable way. The gating strategy makes each block in an LSTM network behave like a memory unit and thereby allows the network to decide when and to what extent it should remember/forget the information, as the input is presented to the model at each time step. This is achieved through four single neural network layers (as opposed to the single layer in regular RNNs,) that are connected to each other in a special way (Figure 1.5a). LSTM networks have become popular after being successfully utilized in large-scale tasks with complex temporal state dependencies, such as speech recognition and language translation.

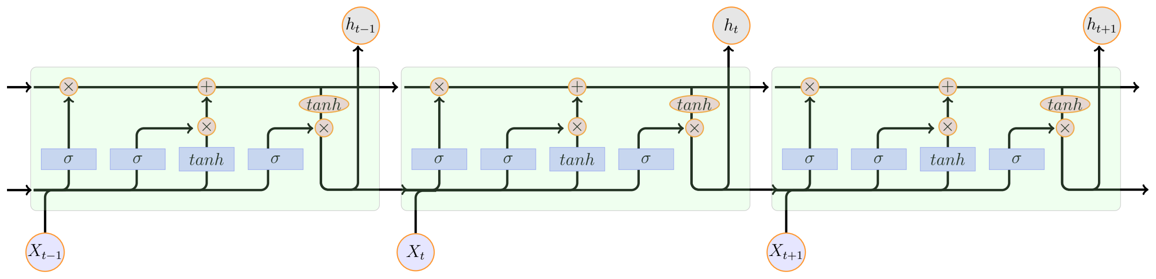
Several variation of LSTM networks have been proposed which differ from each other mostly by their gates and internal connections. Figure 1.5 depicts one popular LSTM architecture which is used in my proposed pipeline in chapter 2. Accordingly, the following equations govern the behavior of each LSTM block,

$$
\begin{aligned}
f_t &= \sigma(W_f.[h_{t-1}, x_t] + b_f]) \\
i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\
o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\
\tilde{C}_t &= tanh(W_C.[h_{t-1}, x_t] + b_C) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
h_t &= o_t * tanh(C_t)
\end{aligned}
$$

where f_t, i_t and o_t are the forget, input and output gates, respectively. The gates can be considered as knobs through which the cell state (C_{t-1}) is selectively manipulated from the beginning to the end of the unit as the signal passes through. The gating mechanism is performed through the sigmoid layers that act on the current input (x_t) and the previous hidden state (h_{t-1}) with their parameters respectively tied across the unrolled units. The output of each gating layer is element-wise multiplied by the corresponding signals to forget, magnify or add relevant information to the cell-state that is subsequently passed to the next unit, all in a differentiable way. Like CNNs, LSTM networks can become deep by stacking multiple LSTM networks on top of each other where each upper network receives the output from the one bellow it as its input.



(a)



(b)

Figure 1.5: Recurrent neural networks. (a) Diagram of the type of LSTM cell used in chapter 2. Small pink circles denote pointwise operations, arrows indicate the direction of information flow, merging arrows represent vector concatenation. (b) unrolling an LSTM network for three time steps.

CHAPTER 2
ENHANCING PREDICTION OF SEQUENCE SPECIFICITIES OF DNA BINDING PROTEINS

2.1 Introduction

DNA binding proteins are key components of different cell processes including transcription, translation, repair, differentiation, and replication. Protein-DNA interactions play important roles in all these processes. One important protein-DNA interaction that is vital for expression of genes in the cells is the interaction between the Transcription Factor (TF) and the DNA. Numerous TFs are involved in regulation of gene expressions. In humans, about 8% of all genes have been marked as transcription factors [18]. Characterization of binding sites associated with such proteins is the first step in building transcriptional regulatory networks. It can also help predict the genes that a TF regulates, annotate them and the genes they regulate, understand regulatory processes in biological systems, identify causal disease variants and, design pharmaceutical drugs that promote or suppress expression of target genes. Unfortunately, exhaustive assays of TFs for all possible short fragments of DNA (i.e. k-mers) is not possible as technology remains a bottleneck. This makes characterization of TF binding sites a complicated task that is still not solved to an accurate level. The current high/mid-throughput techniques, however, are able to provide low-resolution information about where and how specific TFs can bind on DNA. The remaining work rests on the shoulders of bioinformaticians to exploit interpretable and actionable knowledge from the large volume of noisy and fuzzy data generated by such methods. This is of course a complicated task, as will be discussed shortly, and demands advanced computational techniques to partially mitigate the challenges.

In the rest of this chapter I will first discuss some of the major complexities involved in the TF-DNA interactions and then provide a short review of the current approaches that try to address some of these challenges. Our focus will be on the accurate prediction of TF-DNA binding specificity. Then, I will introduce DeeperBind [19], our first approach to solve the problem. To evaluate my models, I will use protein binding microarrays experimental data from a number of TFs that were used by previously published studies for benchmarking. Through analysis and evaluation of DeeperBind, I will show its promise as well as challenges and shortcomings that are not fully handled by it. Subsequently, I expand DeeperBind to a series of models, to address some of those shortcomings and show that the newer models offer significant performance boosts in their prediction power. Finally, I will conclude this chapter by shedding more light on what challenges we successfully addressed, and what remain to be tackled in future work.

2.1.1 On the Complexity of Protein-DNA Interactions

Prediction of DNA-protein binding and specificity involves multiple layers of complexities starting with biophysical interactions in its lowest level to the combinatorial protein-protein and protein-protein-DNA associations. Each level of complexity involves a variety of mechanisms that impact the interactions and the specificity with respect to binding sites. Here, I review a few (among many) such complexities that affect the binding forces between proteins and their DNA motifs and argue that in order to develop an accurate model that can be practically used for predicting binding specificities, these complexities should be considered either directly or indirectly [20].

1. DNA binding site degeneracy. As opposed to proteins that specifically bind to DNA (such as restriction enzymes, where a consensus sequence can be used to characterize their binding sites,) transcription factors can bind to a wide range of sequences, although with different affinities. The level of this affinity in itself can play a role in the regulation of underlying transcription pathways. In fact, it has been shown that some low-affinity sites have been selected by evolution [21] which highlights the im-

portance of the affinity level in such pathways.

To quantify the affinity level, different methods have been proposed during the past years, the most prominent of which is Position Weight Matrices (PWMs). PWMs and its extensions form the basis for many of the more sophisticated approaches that are in use today. Due to its similarity to the convolutional filters, recent successful models from image and speech processing can be readily adapted and used for prediction of binding sites. This serves as the foundation for majority of recent models including DeeperBind and DeeperBind Plus.

2. DNA secondary structure. Proteins do not bind to DNA solely based on the base sequence readout. Sometimes, the mechanism of site recognition is only indirectly related to the DNA sequence (i.e. through the DNA shape). This means, for example, that DNA sequence can create a conformation (such as a kink in the DNA due the sequence or a change in DNA groove width [22]) that changes its preference to the binding proteins [23]. What makes this a challenge for predictive models, is the fact that this structural change is the result of a distributed contribution of different base-base and base-residue biophysical interactions that cannot readily be incorporated into the models, or at least it is hard to do that in a systematic way. Our approach presented in this chapter partially addresses this challenge due to its ability to capture temporal (i.e. sequential) information.

3. Docking geometry of protein and DNA. The formation of energetic interactions (such as hydrogen bonds) largely depends on the relative orientation of the key protein residues and DNA nucleotides [24]. This means that different protein folds result in different degrees of interactions as the same residue-base will not have the same level of contact. Besides that, even proteins that are structurally homologous can yield different docking orientations. Moreover, the docking geometry is also dependent on the DNA sequence, therefore, same binding sites located in different genomic con-

texts may exhibit different preferences due to the same reason. Because the models that we are concerned with in this thesis only consider the raw DNA sequence, they can't model this type of complexity directly. However, if the model is comprehensive enough it can learn such patterns indirectly. In particular, deep-learning based models (such as the one we present in Sections 2.3 and 2.7) are known for their high capacity and can potentially learn some of these patterns, although in a non-interpretable way.

4. Impact of non-binding DNA sequence. Even though binding is the outcome of interactions between DNA proteins and the DNA binding sites, the DNA flanking these sites may affect the site's affinity [25]. Such DNA bases may or may not be immediately adjacent to the site. What makes these flanking bases different from those that are part of the binding site is that their contribution is not through a contact with TFs, but rather, through changing the 3D structure of the binding site itself. Therefore, such a complication cannot be dealt with by expanding the models core to accommodate a larger k-mer, rather, it demands a different treatment. In our proposed models in Sections 2.3 and 2.7, we do not explicitly encode such a complexity into our model, however, if this behavior is well represented in the high-throughput experiment data, then the developed model has the capacity to pick it up and encode it into the learned model, although again in a non-interpretable way.

5. TFs exhibit different binding modes. Massive amounts of data generated by high-throughput technologies have elucidated another level of complexity, the protein binding mode. This difference in binding mode can result in different binding preferences that are chiefly different from each other (hence, multiple consensus sequences). Such complexity can cause deviation from the optimal parameterization of models (esp. models that rely on PWMs), if they do not account for it. Different binding modes can be due to multiple reasons, such as proteins having multiple binding domains or the DNA motif being composed of two parts (i.e. bipartite mo-

tifs [26]) separated by a variable number of bases. Multiple studies (e.g. [27]) have shown that different binding modes is in fact a prevalent phenomenon, and therefore, is worth being encoded into new models. The model presented in this thesis can potentially capture such a complexity due to being trained on probes (instead of k-mers) that are large enough to accommodate multiple potential motifs located near each other.

6. Other confounding factors that play a role, in-vivo. From the computational perspective, perhaps the most complicated interaction to capture is the formation of multi-protein complexes that takes place in a timely manner. Such interactions cannot be measured in-vitro and yet can result in significant diversity in the binding preference of the resulting complex. The other challenge in learning such patterns is that all generalizable models that we know of, often consider TFs in isolation and the role of co-factors are disregarded, which can lead to an improper training trajectory. Another confounding factor that is well worth investigation is the role of TFs' splice isoforms. More than a thousand of about 1400 human TF genes have identified isoforms. Such isoforms can potentially change the binding affinity if not the motif. Last but not least, TF-DNA binding is conditioned on the level of TF expression itself, as well as chromatin accessibility in the cells. Unfortunately, the analyses presented in this thesis falls short in finding such a relation as it only uses the DNA sequence for training. For future work, one may consider inclusion of such evidence as external information to better predict specificity.

While design and training of a model that accounts for all such complexities requires a variety of data modalities with relatively high signal-to-noise ratio (and in high quantity), external information, and a significant amount of computational resources, some of the above mentioned challenges can directly or indirectly be accounted for, by using advanced techniques in machine learning and using only DNA sequence data and of course sufficient amount of data for training.

The rest of this chapter is organized as follows. In Section 2.2 I present an overview of methods that are frequently used for modeling the binding specificity of TFs. Then, in Section 2.3 I introduce our first approach, DeeperBind [19], for prediction of protein binding specificity using DNA sequences comprising potential motifs and flanking regions. An analysis of the observations and a benchmarking of the method will be presented in Sections 2.4 and 2.5, followed by a discussion of the shortcomings and challenges that were not accounted for properly, as discussed in Section 2.6. Subsequently, by taking into account the observations and results of our analysis in the previous section, in Section 2.7 we expand DeeperBind to a series of more powerful ones, collectively named DeeperBind Plus, and show that DeeperBind Plus models offer significant improvements over the former model. Finally, we conclude this chapter by shedding light on future works that can lead to further improvements in future models.

2.2 An Overview of Models for In-Vitro Characterization of Transcription Factors Binding Specificity

To date, a number of computational tools that are applicable to data from in-vitro experiments have been developed to model binding specificity of DNA binding sites (see [28, 29] for a comprehensive assessment of some of the most popular approaches in the pre-deep-learning era). Most of these methods are based on classical techniques that, while differentiated by their underlying learning model, also differ chiefly by the objective they optimize. Different objectives include rank statistics (e.g.[30]), likelihood of some probabilistic model (e.g. [31]), Spearman correlation between the measured and the predicted intensities (e.g. [32]), the emission probability for a given sequence according to a Hidden Markov Model (HMM) (e.g. [33]), or a regression loss (e.g. [34]). These models often fall short in accounting for different levels of complexities that play a role in the binding process and as a result are oversimplistic. Here I briefly introduce some of the most prominent methods that are applicable to in-vitro experimental data and have been used as baselines in similar studies [35, 31, 33, 36]. I will use these methods later when evaluating our proposed models in this chapter and the following one.

2.2.1 Position Weight Matrices

The concept of a Position Weight Matrix (PWM) was introduced in 1982 as an improvement to the more limited model, *consensus sequence*. Although very simple, PWM remain the most prevalent way of characterizing motifs. While it is not per se suitable for prediction of binding specificity, PWM or its variants are the integral building blocks of most (if not all,) advanced models for prediction of binding specificity of TF-DNA interactions. PWMs can be easily learned on data generated from a variety of technologies, including the PBM.

A PWM is described with a 2D matrix of four (20 in case of proteins) rows and N columns corresponding to each of the N positions, in order, in the motif being studied. A PWM matrix is calculated through the auxiliary position frequency matrix, M , that counts the number of occurrences of each nucleotide (amino acid), S_i, at position i of the sequence S. PWM also account for the background distribution of nucleotides (residues) by dividing the probability $P(M_{ij})$ by this prior b_j . This is to give more weight toward deviations from the background model, for example, a "T" in a high GC content sequence should have a magnified contribution to the overall score. PWMs are often log normalized to avoid numerical instability and promote more efficient computation of sequence scores.

A PWM by itself is very limited due to the underlying assumptions that do not always hold true. Most notably, PWM assume an independent contribution of individual motif base pairs to the overall affinity. On the other hand, extensions of PWM using neighboring dinucleotides and trinucleotides are not able to fully model the overall dependencies, which in any case are not limited to the neighboring nucleotides. Furthermore, PWM assume a linear relationship between the sequence base pairs and the specificity, which is not accurate. The existence of low-affinity sites renders PWM an infeasible approach especially if such sites deviate markedly from the binding site's consensus sequence. This spurs development and use of more advanced methods that have the potential to address the shortcomings.

2.2.2 Seed-and-Wobble

Seed-and-Wobble [6] was designed by the inventors of the PBM technology, for characterization of binding motif and specifically for PBM data. This makes the method an interesting baseline candidate for our benchmarking task.

Seed-and-Wobble starts by assigning each 8-mer with up to 3 gapped positions a score called the enrichment score, according to the following rank-sum based statistic:

$$E = \frac{1}{B+F}[\frac{\rho_B}{B} - \frac{\rho_F}{F}] \tag{2.1}$$

where B and F are the sizes of background and foreground k-mers and ρ_B, ρ_F are the sums of background and foreground ranks, respectively. To find a more compact representation of motifs in the form of a PWM, the following steps are performed. First, the enrichment scores for each 8-mer are averaged across the two replicate arrays. The 8-mer corresponding to the highest score is then used as a "seed". Then for each of the 8 nucleotides in the seed, all four variants are inspected by considering the ranks of the probes that include them. This gives each variant a normalized score. Then, each seed can be expanded if addition of another position to it increases the overall score, meaning that some positions in the original seed have a small information content and hence can be dropped and new bases added to the seed. The number of gaps should not exceed 3. Finally, the scores computed for each position can be used with a scaled Boltzmann distribution to generate sequence logos. It is worth noting that this method takes into account all features in the array and not just the top positive and negative ones according to some threshold. This helps in extraction of more useful information that is distributed among all probes.

2.2.3 BEEML-PBM

Binding Energy Estimation by Maximum Likelihood for Protein Binding Microarrays (BEEML-PBM) [34] is another TF binding specificity prediction model tailored for the PBM generated data. The goal was to improve the performance of Seed-and-Wobble by two major changes. First, BEEML-PBM is a regression model that looks for an energy model for a PWM that best describes the motif. The authors for the method maintain that the majority of TFs that are claimed to be represented with two or more PWMs are in fact due to the wrong parameterization , so one PWM, if trained properly, can explain most of the intensity variation in the experiments. Second, BEEML-PBM integrates three factors into the underlying quadratic cost function that they optimize. These factors include binding position on the probe, binding probability to each position of the probe, and the PWM energy model. This method uses significantly fewer parameters than the Seed-and-Wobble model and has been shown to surpass it in performance.

Like the previous model, BEEML-PBM does not model the contribution of flanking sequences and in its most parametric form is too simple to be able to capture hidden nonlinear relations between bases that are far from each other. In fact, its energy model assumes independent contributions from each base in the motif.

2.2.4 Rankmotif++

Rankmotif++ [31] was the first model that was trained using relative preferences between intensities, rather than learning the intensities directly. This has the benefit of being more robust to the noise in the data and other confounding factors that may not be accounted for directly. Rankmotif++ is a probabilistic approach that learns the weights for a PWM such that the likelihood of observed relative preferences between probes is maximized. The model parameters (which can be the weights of a PWM and a scaling factor) in Rankmotif++ are learned such that the distance, according to the model, between any pair of probes i, j with $INT(i) > INT(j)$ translates to a probability measure. In other words, the higher the distance, the higher this probability. To make the optimization tractable, they choose a parametric form for the cost function. While the authors did not mention it, in the general form, this can be casted into a Siamese network [37, 38] that uses only one PWM for computing the affinity scores. Results of the performance comparisons show that this model outperforms all the above-mentioned models.

2.2.5 Kmer-HMM

Kmer-HMM [33] uses a hidden Markov model to represent multiple motifs from PBM experiments. K-mers are sorted by the median intensities of the corresponding probes. The top k-mers are then culled and aligned. Then, an HMM is trained over the alignment matrix using the well-known unsupervised Balm-Welch algorithm. Once trained, the score of a given K-mer can be calculated using the popular Viterbi algorithm [39]. To find motifs, the most probable state paths can be found, using the N-Max-Product algorithm. The key contribution of Kmer-HMM can be attributed to its ability to represent multiple motifs, however, it does not make the right use of the signal intensities. In fact, signal intensities are only used in the first step for filtering out low-score k-mers. Moreover, Kmer-HMM only trains on a very small set of positive probes/k-mers and does not make use of the information that is readily available (although noisy) in the rest of the dataset.

2.2.6 DeepBind

DeepBind [36] was the first deep convolutional method designed to address the need for accurate characterization of TFs motifs. Despite its name, it employs only one convolutional layer followed by non-linear thresholding, a max layer (instead of max-pooling) and one/two fully connected layer(s) serving as a regressor for estimating the TF binding affinity with respect to a given probe. Therefore, DeepBind computes a binding score according to the non-linear function, f, built from composition of other non-linear functions[1]

[1]In short, $f(s) = net_W(pool(rect_b(conv_M(s))))$

in an end-to-end trainable way. In the convolution stage ($conv_M$) DeepBind scans DNA sequences by applying a number of $4 \times m$ motif detectors, M_k. The motif detectors are much like PWMs of length m but without requiring the coefficients to be log-odd ratios. Note that in order to be applicable to varying length sequences, the pooling layer has to take the maximum over the complete sequence so that the fully connected layer receives inputs of consistent dimensions. The method utilizes dropout regularizers as a strategy to combat overfitting. Since the final solution highly depends on the weight initialization step, multiple networks are trained and the best one is used for prediction. To the best of my knowledge, DeepBind used to be the most accurate published model for prediction of TF-DNA affinity from sequence data only.

Despite its clever design, DeepBind suffers from three important shortcomings listed below:

1. The key shortcoming is that it lacks the ability to capture the temporal dynamics hidden in DNA sequences. This is due to the max operation that is performed over all the nucleotides. In other words, instead of using a max-pooling layer, which is an integral part of image-based Convolutional Neural Networks (CNNs), DeepBind computes the maximum of the convolution scores across the entire probe. This is due to a hidden limitation of the model that input probes cannot be resized before being fed into it, much like what we do when we rescale 2D images. As a result, and in order to be able to predict on varying length DNA sequences, DeepBind has no choice but to take the maximum of feature maps across all positions. This limits the model from learning temporal information that exists in positional feature maps. The other implication is that if there are two (or more) motifs in an input probe, DeepBind only considers the one with the highest score and cannot model their combined effect when predicting probe intensities. In fact, it is not an unrealistic scenario to have multiple mid-affinity motifs leading to a relatively high binding score, if we consider that TFs can potentially bind to any sequence at certain specificities. Therefore, to

enable more accurate predictions it makes sense to consider the contribution from all possible k-mers in a holistic way. Furthermore, this is not just a limitation in modeling, but rather, this simplification may bias the training stage and can lead to a sub-optimal trained model.

2. DeepBind models a binding score directly which makes training such a network a challenging task. This is for the most part due to other confounding and contributing factors that are not present or represented in the DNA probe and yet change its binding affinity in an unknown, non-linear way (see Section 2.1.1). As a consequence, direct modeling of the binding intensity in PBM data leads to sub-optimal training and lacks robustness.

3. As discussed before, the affinity between DNA and TFs may depend on other factors such as flanking DNA bases that may affect DNA shape downstream/upstream of the motif which facilitates binding on DNA. In such cases, neither the flanking sequence nor the main motif exhibit a high binding score but the combination of the two yields a synergistic effect. Therefore, it is necessary to account for both binding sites and signals around them, through effective modeling of the binding's temporal aspect.

In the rest of this chapter, I first propose a new model, that integrates recurrent networks to learn a more complex pattern from convolutional feature maps to mitigate the first and the last failures from the above list. Then, in Section 2.7 I will address the second shortcoming by extending DeeperBind to DeeperBindPlus and its variants. I will evaluate the proposed models on data from a few previously published PBM data and compare their performance with the baseline models introduced above. I conclude this chapter by discussing the strengths and more importantly, the weaknesses of such models and what needs to be considered for further improvements in future.

2.3 DeeperBind

DeepBind and DeeperBind are both inspired by the record-breaking performance of deep convolutional neural networks in image processing and vision domains. This becomes more interesting if we notice the analogy between the convolution operation on images and applying the PWM to DNA sequences. In other words, DNA sequences are just one dimensional images. Pixels in a 2D image are dependent on neighboring pixels as well as the content of the image (e.g. whether the image is a dog or a tree). Likewise, nucleotides in a DNA sequence depend on neighboring ones, especially on a genome's functional features, as well as functions of the loci that host such features (e.g. whether it is part of an exon or intron). The differences being that the input in the former case is two dimensional with often three channels (corresponding to the R, G, and, B colors) while in the latter case we are dealing with one dimensional sequences of four channels (corresponding to A, C, G, and T nucleotides). Another minor difference is that the channels in the former, while dependent on each other, can take any value in the valid range of pixel intensities, but in the latter, each channel is a one-hot coded vector of size four. Despite these differences, the two operations are analogous. This suggests that we might benefit from the same performance boost as we have witnessed in the image processing domain, which in fact turns out to be true. The biggest success of using 1D convolutional neural networks comes from the ability to incorporate multiple convolutional filters into one model in an end-to-end trainable way, in light of the fact that a single PWM per se is not expressive enough to capture all the binding dynamics, as discussed before.

Figure 2.1 illustrates the architecture of DeeperBind [19]. As a preprocessing step, the nucleotide sequences of PBM probes are converted into a three dimensional tensor of size $b \times L \times 4$, where is b is the batch size and L is the maximum length of probes excluding the primer. In case of PBM, this is set to 35 base pairs. In the next step, the encoded input is fed into a convolutional layer with a k filters of size $M \times 4$ where M is the filter size. Using these filters, this layer performs k one-dimensional convolution on the input probes. This generates an output feature map of size $b \times (L - M + 1) \times k$. For each slice of this tensor that corresponds to sample $i \in \{1, \ldots, b\}$, in a given batch, and probe position $j \in \{1, \ldots, L - M + 1\}$, this layer computes a latent representation that characterizes positional affinity with respect to each of the k kernels. The key step left is to learn the temporal dynamics of these feature maps across the probe. This is done through the next layer of our model, which is a 2-layer LSTM network. When unrolled, each cell in the first-layer LSTM network receives the feature maps generated for one position and utilizes its inner network to extract new useful information and manipulates the currently passed information from the previous layer and pass the resulting tensor to the next cell. Meanwhile, the cell's hidden state is also given as input to the LSTM layer above. LSTM has the potential to learn non-linear temporal patterns from the feature maps that the CNN module learns. While CNNs can effectively capture individual motif features, LSTMs are able to learn and relate distant features that a CNN extracts. Therefore, the combination of these two key techniques play a complementary role through extracting more informative features that can be used by any classical classifier for prediction of the binding affinity. As a final step, we can input the output from the last cell in the last layer into a classifier, such as a fully connected neural network for decision making. This output encodes all the temporal and local features from the beginning of the probe to the end and can therefore be used for decision making. We used a two-layer fully connected layer with dropout regularization to predict the normalized signal intensities from the PBM experiments. Finally we optimized model parameters using ADAM [40] optimizer (with an exponential weight decay strategy) which compared to other optimizers resulted in more accurate model in smaller number of iterations.

We used tensorflow [41], one of the most popular libraries for prototyping neural network models, to develop, learn and infer the scaled intensities. Furthermore, in order to make a fair comparison between DeepBind, DeeperBind, and DeeperBindPlus, we used

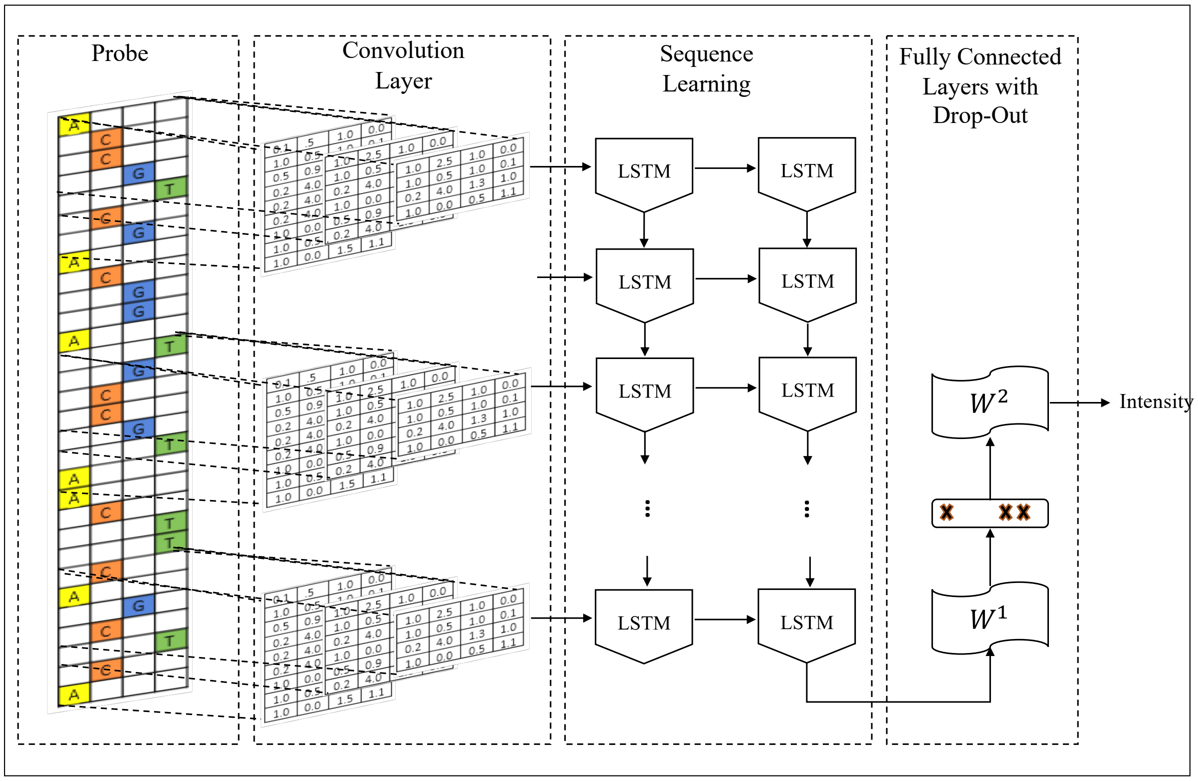


Figure 2.1: Diagram of DeeperBind's model architecture.

the same library to implement each method.

2.3.1 Data Preprocessing and Parameter and Hyper-Parameter Optimization

PBM Data

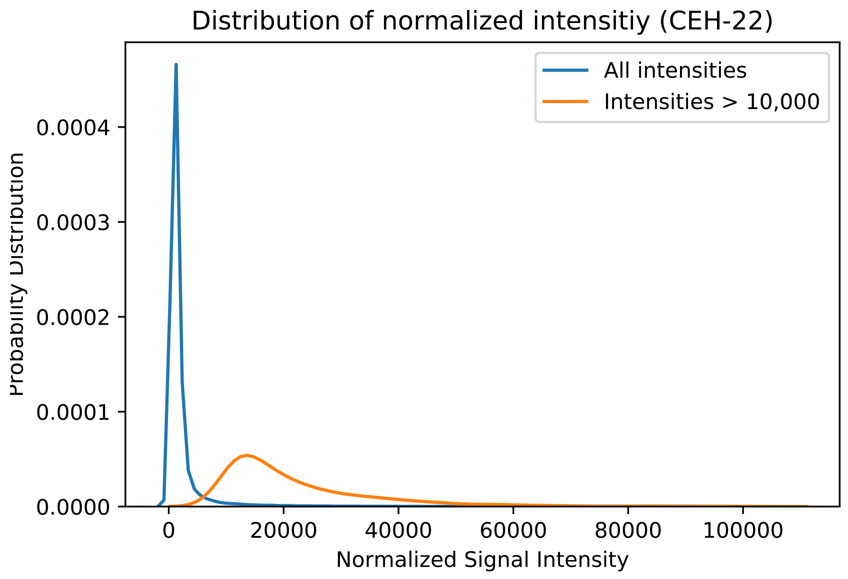
We retrieved PBM experimental data, previously published in [6, 42], from the Universal PBM Resource for Oligonucleotide Binding Evaluation (UniPROBE) database [43]. We used probe sequences and the normalized measured intensities, and dropped probes that did not pass quality control tests performed by the authors. Each probe in the UniPROBE database has length 60 nucleotides. The first 25 nucleotides, located at the 3' end of the sequence contains a single thymidine linker attached to a glass substrate, followed by a constant 24-nt primer and is shared between all probes. The remaining 35 nucleotides are variable length sequence that we use for training our model.

The normalized intensities reported by PBM experiments usually fall in the range $[0, 10^6]$. While the intensity values per se do not have any biological meanings, the relative intensities do represent how specific a probe binds to TF, compared to other probes. In fact, the scale of normalized intensities can vary between replicate experiments. This also suggests that perhaps instead of learning the intensity itself we should learn some relational distance between the inputs[2]. Nonetheless, learning the signal intensities directly is also a legitimate way for analysis of binding affinity. Figure 2.2 shows the distribution of measured signal intensities from PBM arrays for two TFs, namely, CEH-22 and Cbf1. The distribution in both cases show a very long tail. Along with the main curves, the intensity distribution for the top 10% of probes is also plotted. These are mostly the probes that host moderate-high affinity motifs. Since our goal is to model the underlying motifs, we compute the z-score of the intensity and use it as a rescaled score for our regression model. For those TFs where replicate arrays exist, I used one replicate for model fitting and the other for evaluating the trained model. Note that, to make an unbiased evaluation, these arrays were designed with different de Bruijn sequences. For model fitting, I split the training array into training and validation sets by randomly selecting 0.85% and 0.15% of the probes, respectively.

On Reproducibility of PBM Experiments

PBM experiments provide a setting that allow us to interrogate the direct interactions between a TF and putative sites. This means the impact of other co-factors and confounding elements are eliminated. Even though this by itself can be considered as a restriction, nonetheless, it reduces some layers of complexities that would otherwise mislead model optimization. In spite of that, learning the existing binding patterns still appears to be challenging, in part due to the measurement nosie and batch effects and partly due to expressiveness of models and how they are trained.

[2]We will revisit this question later in this chapter.



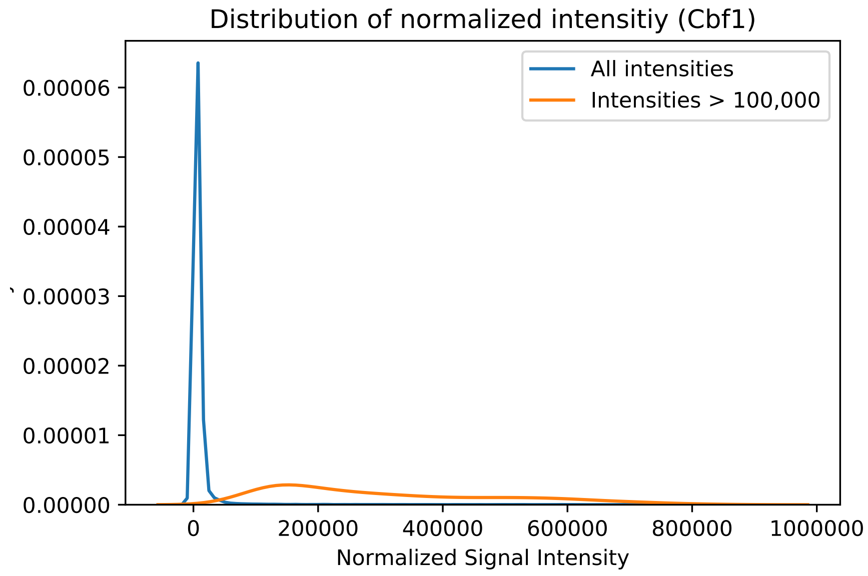


Figure 2.2: Normalized intensity distribution for transcription factors CEH-22 and Cbf1 .

Table 2.1 shows the Spearman correlation of median 8-mer intensities between the two arrays. Each array contains measured signal intensities from an independent assay (possibly with a different machine). We computed this statistic for two cases, when all 8-mers are present, and when only the top 100 8-mers are included. From this table, it becomes evident that, the correlation drops as more k-mers are included. This is especially true for

Table 2.1: Spearman correlation between median intensities of k-mers between the two arrays

TFs	All 8-mers	Top 100 8-mers
Cbf1	0.48	0.83
CEH-22	0.47	0.55
Oct-1	0.60	0.76
Rap1	0.20	0.50
Zip268	0.28	0.79

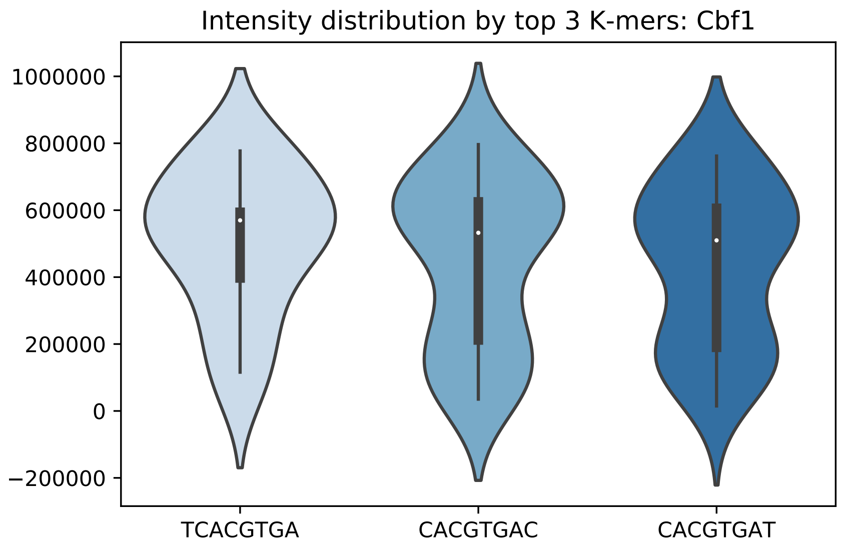
Rap1 and Zif268[3]. Furthermore, we can use the top k-mers for validation of models as they tend to be more reliable. This also puts an upper bound on the performance we should expect from any de novo model that learns only from k-mers (in this case k=8)[4]. Of course, as I will note shortly, we will exploit more features from the probes, and hence, we are not bound by this limit.

The next question that we should address is the reliability and reproducibility of the experiments with respect to the top probes. Figure 2.3 depicts the violin plots of probe intensities that contain each of the three top 8-mers or their reverse complements. The top 8-mers where selected based on the median[5] intensities of those probes that contain them (or the reverse complement of those 8-mers). We selected the top k-mers to mitigate the background noise that is present in PBM measurements, as discussed above. This is also consistent with previous findings [34]. We did this analysis for TFs Cbf1, and Oct-1. The distribution for the other proteins follow a similar pattern. According to this figure, even k-mers that are well represented by a motif model can exhibit a wide range of binding affinities. This could be in part due to the batch effect and other measurement noises, but also other factors that relate to the features of positions upstream/downstream of the sites. DeeperBind targets these extra features to boost the prediction performance.

[3]These five TFs were used for analysis in [6].

[4]Under the assumption that the de Brouijn sequences that are in the two arrays are independent of each other, and within each array, randomly distributed.

[5]Median is used to counter the available noise.



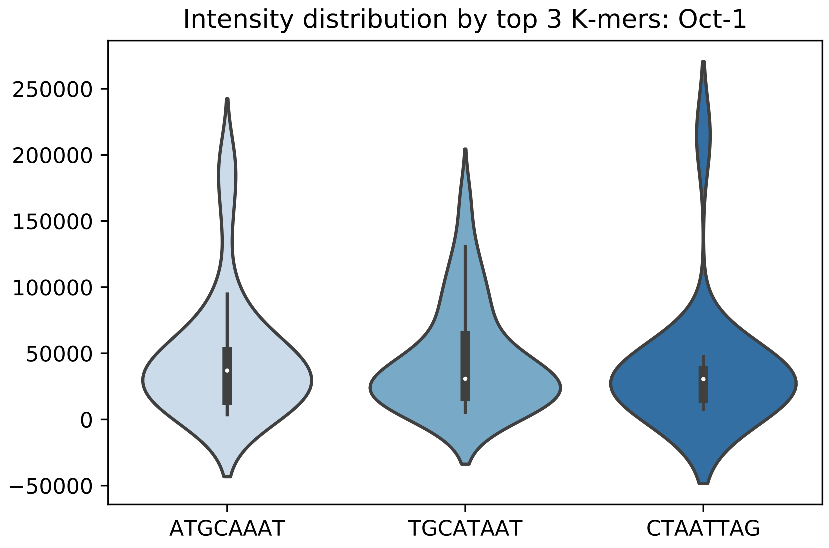


Figure 2.3: Violin plots of intensities for the top three 8-mers. Top: Cbf1 and bottom: Oct-1.

2.4 DeeperBind: Experimental Results and Analyses

I retrieved PBM data for five transcription factors (i.e. Cbf1, Rap1, Ceh-22, Zif268, and Oct-1) that have been analyzed in [6]. These proteins belong to diverse structural classes and from different organisms. Cbf1 and Rap1 both belong to the yeast yeast *Saccharomyces* *cerevisiae*. Cbf1 has a basic helix-loop-helix domain which binds the motif with consensus sequence CACRTG[6]. Rap1 is a DNA binding protein with a Myb domain that acts as both activation and repression, depending on the genomic context. Ceh-22 is from the nematode *Caenorhabditis elegans* and has an NK homeodomain whose binding consensus sequence is NNVYACTYVW. Zif268 is a mouse protein with a Cys2His2 zinc finger domain, and finally Oct-1 is a human POU homeodomain transcription factor. For each of these proteins, data for two different PBM experiment exists in the UniPROBE database [43, 44]. Table 2.2 shows binding motifs found for each protein using the BEEML-PBM method [34].

2.4.1 Parameter and Hyper-Parameter Optimization

We conducted a partial grid search for finding a semi-optimal set of hyper-parameters. We searched for different combinations of number of LSTM layers ($\in \{1, 2\}$), LSTM layers sizes ($\in \{10, 20, 30\}$), and optimization learning rates ($\in \{0.0005, 0.001, 0.005, 0.01, 0.05\}$). Furthermore, we tried ADAM and Stochastic Gradient Descent (SGD) optimizers with different batch sizes ($\in \{40, 100, 400\}$). For the DeeperBind and DeeperBindPlus models, we also searched for the optimal number of fully connected layers ($\in \{1, 2, 3\}$) as well as the size of each layer ($\in \{10, 20, 32, 70\}$).

Once we found a semi-optimal set of hyper-parameters, we trained the model 10 times (120 epochs each,) and recorded the best performing model according to the validation set. This is because the initial model weights, which were first set by a Xavier initializer [45], can influence the learning trajectory. Some initializations may drive the trained model towards a local optimum that is not biologically sound. Upon successful training, the performance trajectories for the validation sets very well matched those of test sets, although they were usually higher on test sets, in part due to some degree of overfitting but also because the outcome of each assay can depend on complexities discussed before.

The models were trained and benchmarked on a Deep Learning Machine with a 3.5GHz

[6]By convention, Y and R in consensus sequences represent pyrimidines and purines. N means any nucleotide

12-thread Intel core i7-7800X CPU and an NVIDIA RTX 2080 GPU[7] with a native installation of Tensorflow 1.13.

2.4.2 On the Learning Capacity of DeeperBind

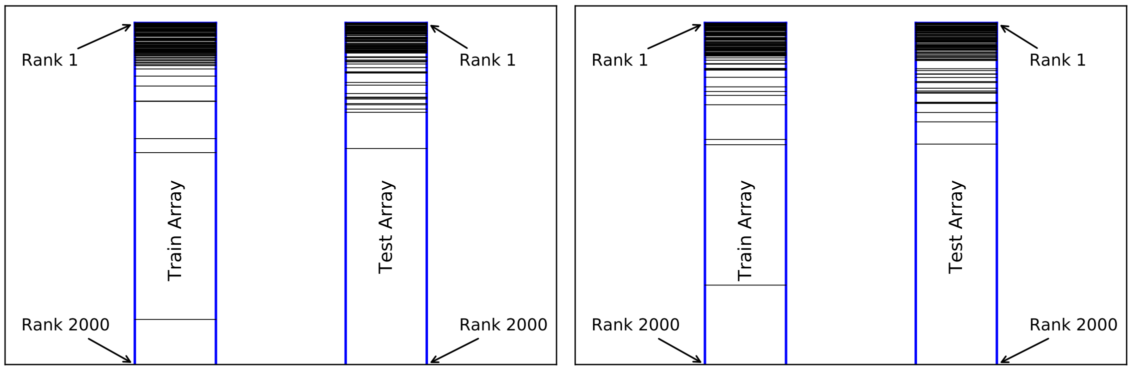
The high number of probes assayed by PBM in a single experiment allow us to design models with increasingly high capacity without being worried too much about the overfitting problem. In light of that, we compared the capacity of DeepBind and DeeperBind in learning the underlying regression tasks. We used the first array to train the two models and then predicted the binding preference of each TF with respect to the top 100 probes, in the same array. The predicted preferences and the normalized intensities of the probes were then ranked and visualized in Figure 2.4. In this figure, the height of each line represents its predicted binding preference rank. Predicted ranks above 2000 are not visualized. The line at the very top of each bar corresponds to the probe with the highest predicted score. In the ideal scenario where all predicted ranks fully match those derived from the normalized intensities, we should have all 100 lines occupying the top 100 positions in each bar. The left bar in each box corresponds to predictions on the training array and the right one shows the ranks when the trained model was applied to the test array.

By comparing the plots of DeepBind and DeeperBind, it becomes evident that our model has a higher capacity to learn complex relations from probe sequences. This is especially the case for CEH-22.

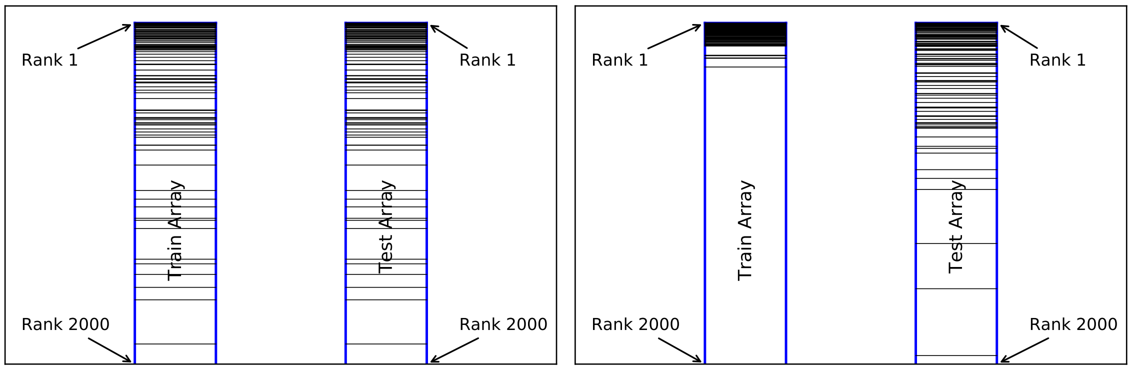
2.4.3 Scatter Plots and Pearson Correlation Coefficients

In the previous experiment I showed that our model has a higher capacity to learn intricate relations. However, having a complex model raises concerns and questions about its generalizability. In this section and the next one, we investigate the performance of our model and the baselines on test sets. We used one array for training and validation and the

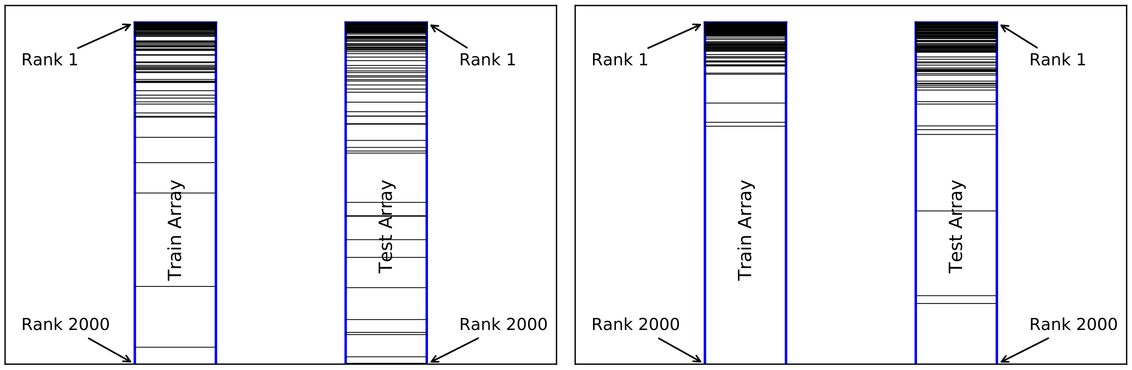
[7]Our benchmarking showed that running the code on 6 cores where at least as fast as running on an RTX 2080 GPU.



(a) Cbf1



(b) CEH-22



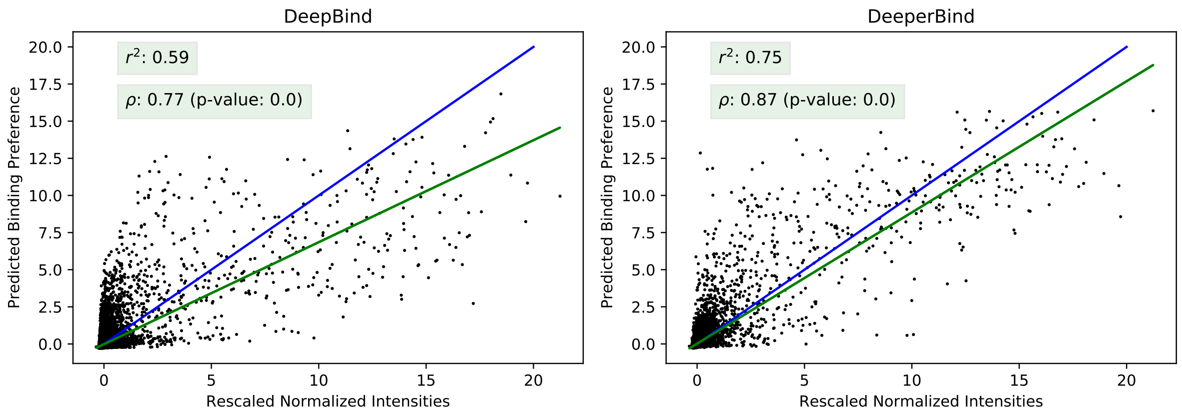
(c) Zif268

Figure 2.4: Comparison of predicted ranks of top 100 positive probes between DeepBind (left column) and DeeperBind (right column) for Cbf1, CEH-22, and Zif268. The left bar in each box shows predictions on the training array and the right bar visualizes predictions when we apply the trained model to the test array. Only ranks between 1 and 2000 are visualized.

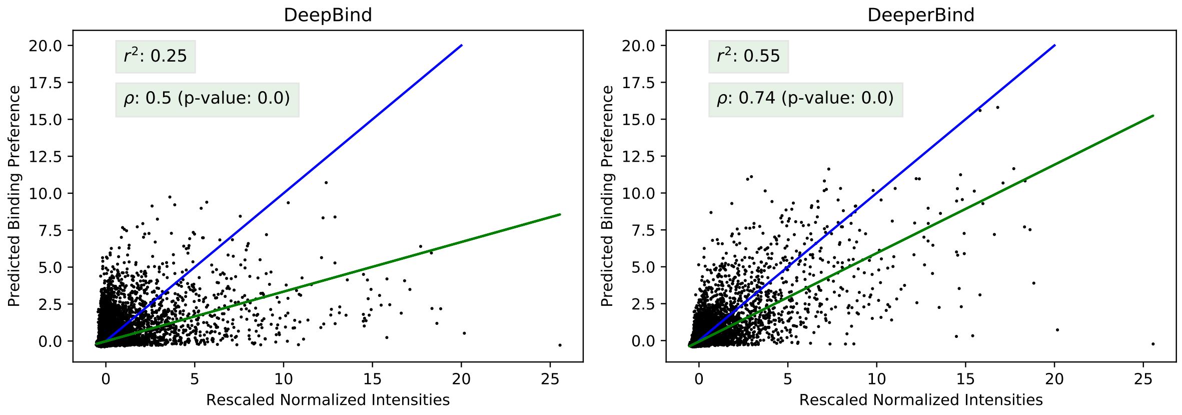
other one for evaluating the performance. Figure 2.5 shows the scatter plots of scaled normalized intensities (x-axis) vs. to the predicted binding preferences (y-axis) for DeepBind and DeeperBind. The blue lines indicate the identity relation and the green ones are the regression lines. r^2 is the coefficient of determination for the regression and ρ indicates the Pearson correlation coefficients. Here, we show the plots for only Cbf1, and CEH-22. Plots of other TFs follow quite similar trends. From these figures and statistics, we see the estimates of the binding preference by our model are closer to the measured intensities, which indicates that despite having more parameters, DeeperBind does not severely overfit the training data. We will make a more comprehensive comparison of the performances between multiple baseline methods, in the next section.

2.4.4 Spearman Rank Correlation Coefficients

In this section I describe a comprehensive benchmarking of all combinations of seven different methods, using the two arrays for each of the five proteins described before. For each TF, we trained methods on one array and tested on the other, and vice versa. The two arrays correspond to different assays with different de Bruijn sequences, which may have been performed with different machines. I used the criteria discussed in [31] for labeling probes as positive or negative. That is, any probe with measured normalized intensity of greater than or equal to the threshold $m_y + 4\sigma$ will be considered as a positive probe. Here m_y is the median of all probes intensities and σ is defined as median absolute deviation (MAD) of the normalized intensities distribution divided by 0.6745. The first column in this table lists the five transcription factors that have been used for benchmarking of various methods (e.g. [31, 33, 19, 34, 6]. The second column shows the size (in parentheses) of positive sets for each array and columns 3-9 pertain to the computed Spearman correlation for seven methods, as follows. The 8-mer method predicts the binding affinity of each 8-mer based on the median intensity of probes that include it or include its reverse complement, and is similar to a look-up table which has 4^8 parameters. As discussed be-



(a) Cbf1



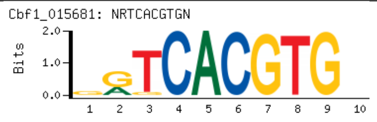
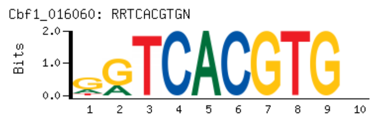
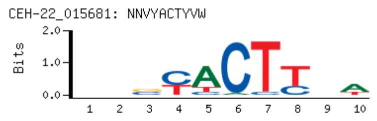
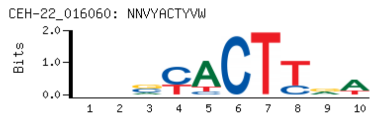
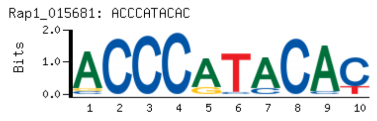
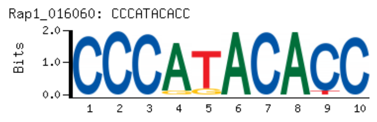
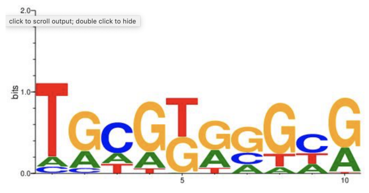
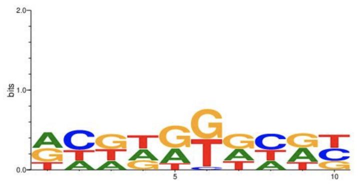
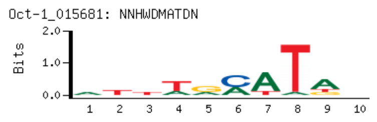
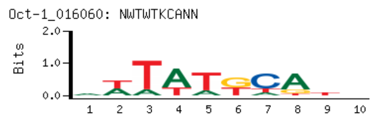
(b) CEH-22

Figure 2.5: Scatter plot of scaled normalized intensities and predicted binding affinities. Blue and green lines indicate the identity and regression lines, respectively. ρ indicates the Pearson correlation coefficient.

fore, the performance achieved with the 8-mer method gives an approximate upper bound on the performance that can be attained using any model that learns a single PWM length 8 (such as models listed in columns 3-5). For Rankmotif++ [31] we trained PWMs of the same lengths as suggested by BEEML-PBM and used option values recommended by the authors. For Seed-and-Wobble [6] and BEEML-PBM [34] we used models trained by the authors for each protein and array. For the Kmer-HMM [33] we used 50 hidden states and k set to 8, as recommended by the program. Finally, for DeepBind [36] and DeeperBind [19], we used five convolutional filters of length 11. For the former, we used a fully connected layer of size 32 neurons and for the latter one, we used two layers of LSTM of size 30, and 20. Because initialization of these networks play an important role in the final solution that these models converge to, we reran each algorithm five times with different initial weights and selected the model that performed the best on the validation set.

Besides Table 2.3 we also calculated and listed the same statistics but applied to all probes, instead of the top ones, to investigate how each method is biased towards positive probes. This is critical, after all, what is important to achieve is how good the model performs on unseen probe sequences. The results are presented in Table 2.4. We also highlighted the best performing model (excluding the 8-mer model) in each table. Several important observations can be made from these two tables. First, DeeperBind is consistently outperforming the other baselines when all probes are considered. Given that the major difference between DeepBind and DeeperBind is in the integration of recurrent networks, we can safely conclude that our model has been successful in modeling temporal signals and accounting for the synergistic contribution of multiple binding sites on probes. Interestingly, DeeperBind also surpasses the 8-mer model. This is because we are not bound by the simplifying assumptions underlying the ground-truth 8-mer model, such as independence of bases on the binding site. We also trained our model with the kernel length set to 8, for an unbiased comparison between the two models and observed that our conclusion still holds. Note that for majority of arrays this does not hold for the DeepBind.

Table 2.2: The five structurally different transcription factors used for benchmarking Rankmotif++, Seed & Wobble, BEEML-PBM, DeepBind, and DeeperBind. Number of positive probes in each array is calculated according to [31]

TF	# Pos Probes (per array)	Domain	Organism	Motif (BEEML-PBM)
Cbf1	#1: 4200	Basic HLH	S.cerevisiae	
	#2: 3294			
CEH-22	#1: 4389	NK homeodomain	C. elegans	
	#2: 3409			
Rap1	#1: 1559	Myb	S.cerevisiae	
	#2: 991			
Zif268	#1: 3116	Cys2His2 zinc finger	Mouse	
	#2: 2801			
Oct-1	#1: 2425	POU homeodomain	Human	
	#2: 3107			

Comparing this table with Table 2.3, we observe a relatively similar trend. In other words, with the exception of Rap1, we are still performing better, with a significant margin. This is a crucial observation knowing the fact that the last two models have been trained on all probes, and not just the set of top ones and yet the trained model generalizes very well by performing well on a biased sub set of the input probes. In the case of Rap1, however, we see a poor performance by both methods. This might be attributed to the small size of the positive probes (compared to the other TFs,) which gives rise to biased sets and hence not representing the datasets well. Hence, reducing the model capacity for this TF can mitigate the problem. Another noteworthy point is that, DeepBind is the runner up model for the majority of cases in these tables. Specifically, when considering all probes, DeepBind often times outperforms the four classic PWM-based models. On the positive sets, however, DeepBind is surpassed by some of the baseline models for Rap1. In fact DeeperBind is also performing poorly due to the reasons discussed above[8].

[8]Since DeepBind and DeeperBind are the best-performer models on our benchmark datasets, in the next sections where I investigate further extensions to such models, I will only include these two models for performance comparisons.

Table 2.3: Rank correlation between predicted intensities for positive (based on the definition above) probes and the binding preferences measured by PBM experiments.

TF	Test Array	8-mer	Rank-motif++	S&W	BEEML-PBM	K-HMM	Deep-Bind	Deeper-Bind
Cbf1	#1 (4200)	0.58	0.47	0.35	0.49	0.54	0.46	**0.64**
	#2 (3294)	0.60	0.50	0.33	0.49	0.56	0.52	**0.72**
CEH-22	#1 (4389)	0.54	0.36	0.23	0.48	0.41	0.49	**0.60**
	#2 (3409)	0.46	0.38	0.16	0.43	0.26	0.49	**0.60**
Rap1	#1 (1559)	0.32	**0.40**	0.36	0.30	0.08	0.28	0.05
	#2 (991)	0.42	**0.43**	0.38	0.28	0.17	0.29	0.08
Zif268	#1 (3116)	0.52	0.39	0.32	0.51	0.32	0.51	**0.60**
	#2 (2801)	0.43	0.31	0.18	0.40	0.29	0.20	**0.54**
Oct-1	#1 (2425)	0.45	0.29	0.26	0.26	0.26	0.32	**0.46**
	#2 (3107)	0.44	0.25	0.26	0.26	0.37	0.37	**0.46**

Table 2.4: Rank correlation between predicted intensities for all probes and the binding preferences measured by PBM experiments.

TF	Test Array	8-mer	Rank-motif++	S&W	BEEML-PBM	K-HMM	Deep-Bind	Deeper-Bind
Cbf1	#1	0.53	0.35	0.27	0.34	0.39	0.44	**0.61**
	#2	0.46	0.32	0.23	0.30	0.35	0.41	**0.61**
CEH-22	#1	0.53	0.29	0.25	0.42	0.44	0.44	**0.54**
	#2	0.43	0.31	0.18	0.34	0.34	0.38	**0.45**
Rap1	#1	0.13	0.13	0.09	0.07	0.07	0.18	**0.33**
	#2	0.19	0.06	0.04	0.05	0.07	0.18	**0.23**
Zif268	#1	0.39	0.31	0.19	0.35	0.44	0.48	**0.59**
	#2	0.36	0.19	0.18	0.33	0.36	0.40	**0.49**
Oct-1	#1	0.60	0.53	0.49	0.49	0.41	0.58	**0.70**
	#2	0.54	0.47	0.39	0.39	0.35	0.53	**0.63**

2.5 Analysis of Inferred Motifs and Positional Bias

One of the key shortcomings of deep networks is the lack of interpretability. While these models solve various problems often with high accuracy, they usually fail to tell us how they solved these problems or at least not in a way that humans tend to think. Models presented in this chapter are not an exception to the rule. Nonetheless, there are some indirect ways to provide some insight into the trained models, especially those that are based on CNNs. In this section, we will answer three main questions, first, what are the best DNA sequences that the trained models favor? In other words, upon learning the binding patterns from data, how can we query our models to activate the learned rules and find the most favorable sequences to the corresponding TFs? Second, how can we extract a motif in PWM form from these trained models? And finally, we will look for any positional bias on the probes with respect to the binding preference.

To address the first question, we followed the strategy proposed in [46] for class model visualization with the difference that instead of optimizing for a score pertaining to a specific class, we are optimizing for predicted binding preferences. To do that, we first fix the network weights that are already adjusted in the first round of training, and then, optimize the input with respect to this score, using the gradient descent optimizer. Note that, we treat the input matrix as a continuous matrix instead of as a one-hot encoded sequence. Obviously we would expect to see the binding motifs well represented in the learned inputs.

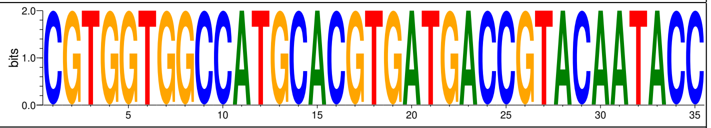
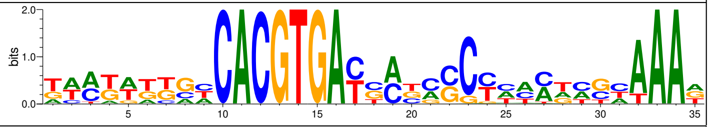
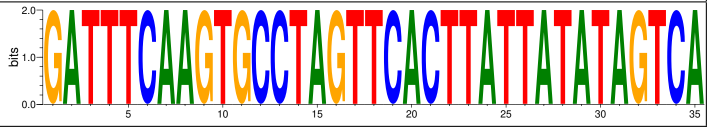
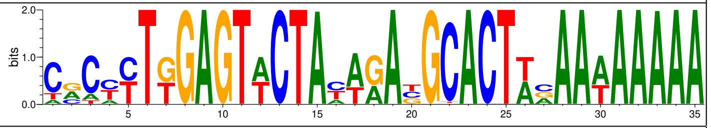
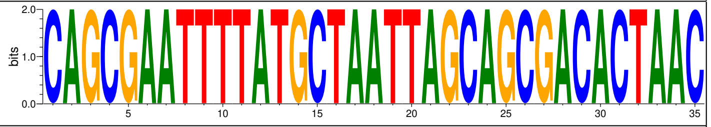
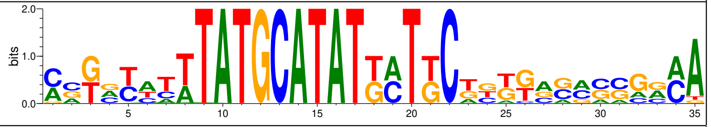
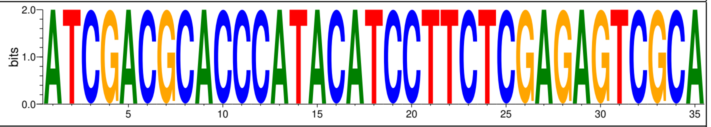
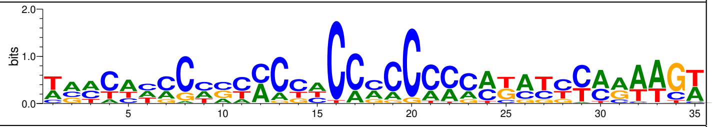
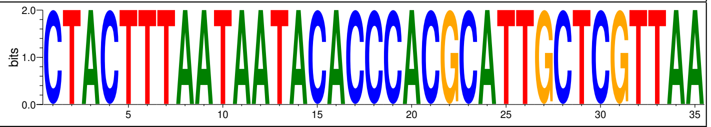
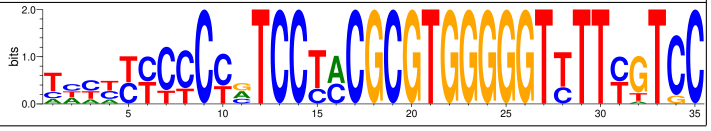
We trained DeeperBind with convolutional kernels of size 8, cloned and subsequently modified it into another network and replaced the input with a trainable matrix of the same size, and then, tuned the input weights of the new network under a new loss, the negated prediction score. To avoid learning a prohibitively large input, we modified the model by truncating the input matrix and then normalizing it across the channel dimension (i.e. nucleotides). This still leaves our model end-to-end trainable while at the same time prevents overweighting the role of individual base positions due to possible large values assigned to the input at those positions. Furthermore, because of the non-convex nature of the optimization task, we repeated the training process multiple times starting from different initial input weights[9]. Table 2.5 shows a pictorial representation of both the highest score probes retrieved from PBM data (first array only,) and sequence logos [47] for generated sequences scored highest, using the above approach. Comparing this table with that of 2.2 reveals a few interesting observations. First, motifs appear to be well represented in the generated sequences. Second, considering both strands, multiple partial (e.g. Rap1) or complete (e.g. Cbf1) replicates of motifs are present in these sequences. These replicates sometime have a lower information content, but they do exist. This proves the ability of DeeperBind in considering the synergistic contributions from multiple existing signals when predicting the binding specificity.

The next important question to consider is whether we can create a degenerate representation of motifs, in an interpretable way. Unfortunately, this is not trivial as CNN-based models integrate multiple kernels in a non-linear way. However, inspired from the experiment elucidated above and with the addition of a regularization term, we can generate the highest scored continuous short input sequence representing a motif of a specified length[10]. A similar idea has been proposed in [48] where the authors optimized a classification score under $L2$ norm penalty and then scaled the resulting network to characterize motifs in a PWM-like format. We followed the same idea with the following changes. First, as discussed above, instead of scaling the input weights as a post-processing step, we modified DeeperBind to normalize the weights, following a truncation layer. Moreover, we masked the input gradients to allow changes to only a short window of a specified motif length. The truncation, normalization, and masking operations are all well behaved and hence differentiable, leaving our model trainable. Furthermore, instead of using an $L2$ norm, we used the

[9]We found 40 to be a reasonable number for repeats.

[10]For each TF we selected the motif length that is the same as the one learned by BEEML-PBM. This makes our comparison less biased.

Table 2.5: Pictorial representation of the highest scored microarrays' probes, and generated probes with scored highest.

TFs	Sequence Origin	Score by Model	Sequence or Logo
Cbf1	PBM	10.64	
	GD	10.84	
CEH-22	PBM	19.37	
	GD	19.62	
Oct-1	PBM	17.15	
	GD	17.50	
Rap1	PBM	1.05	
	GD	1.06	
Zip268	PBM	14.72	
	GD	14.13	

Kullback–Leibler (KL) divergence term, leading to the following optimization formula,

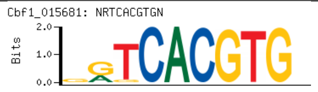
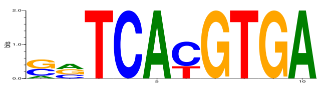
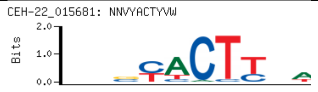
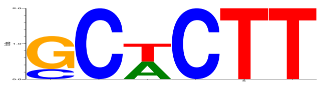
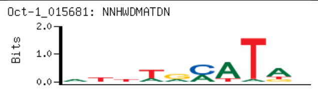
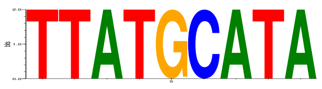
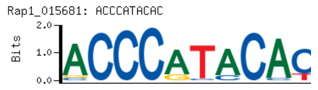
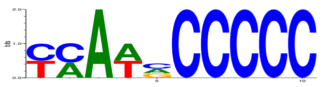
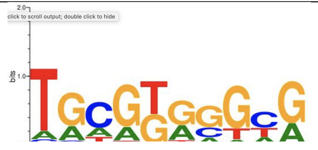
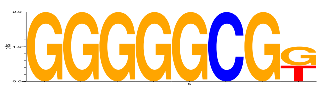
$$\arg\max_{I} S(I_{msk}) - \lambda * KL(I \| P_{unif}) \tag{2.2}$$

where P_{unif} indicates a multinomial uniform distribution over the four nucleotides replicated for all positions, λ is the regularizer coefficient, and I_{msk} is the masked input. To apply the mask, we multiplied the input gradient by the given mask in each step of the back propagation. The KL divergence term ensures that the tuned motif stays as close as possible to the uniform distribution. We initialized the input outside of the masking region with a nucleotide that results in the lowest score and thereby generating a background sequence to start with and positions within the motif range were initialized randomly. The training ends when the improvements over the loss function remain within a small range.

Table 2.6 visualizes the sequence logos resulting from this optimization problem along with the selected motif lengths and positions where they exhibit the highest preferences. Interestingly, while our our predicted motifs are consistent with those learned by BEEML-PBM, there are still noticeable differences. Of course, there is not ground truth motif to compare with and different methods result in different motifs, however, one reason for the differences is that in generating the motifs DeeperBind integrates the role of multiple factors such as motif position, its flanking nucleotide composition, and most notably the contributions from different PWM-like kernels.

Finally, I scrutinized the role of motif location on the binding preference and whether DeeperBind can reflect any such potential positional bias. To that end, I put the learned motifs on different positions of a probe filled with the background nucleotide and computed the predicted binding score. Figure 2.6 depicts this score as a function of position relative to the end of the probe sequence (i.e. the farthest from the glass slide). Interestingly, a universal pattern can be seen from the resulting charts, that is, the score is highest when motifs are positioned on the farther half of the probes. This makes sense as the other end

Table 2.6: Comparison of motifs learned from gradient descent and those learened by BEEML-PBM for five transcription factors. Predicted binding preferences reveals a positional bias.

TF	BEEML-PBM Motif	Gradient Descent	
		Motif	Length (Position)
Cbf1			10 (2-15)
CEH-22			8 (3)
Oct-1			9 (3)
Rap1			10 (10-12)
Zif268			6 (2-20)

of the probes are attached to the primer and then glass slide, which makes less space for the TFs to get engaged with the probe. This pattern is especially pronounced for Oct-1 for which we observe a significant drop in the binding specificity as we get farther from probes' free ends. Another point to make is that DeeperBind has captured this pattern despite the fact that farther positions are closer to the final decision point in the LSTM network and naturally impact the final output more.

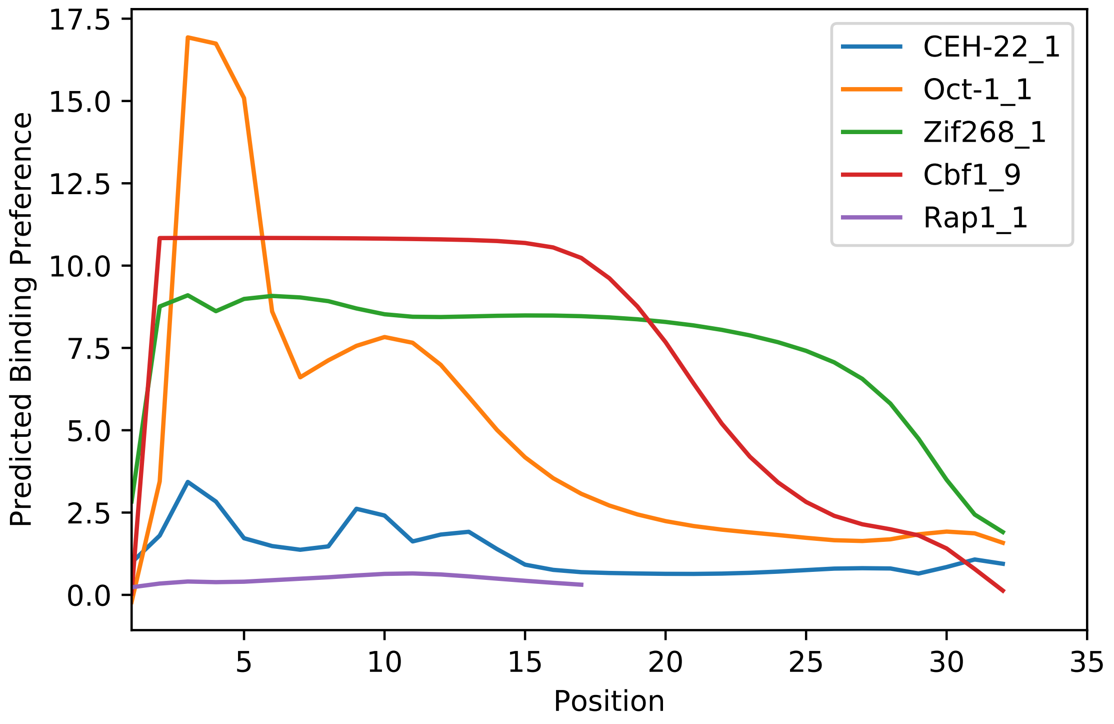


Figure 2.6: Impact of motifs' locations on binding preferences. Positions are relative to the end of probes. Plots show binding is more specific towards the end of probes.

2.6 DeeperBind: Summary of the Results

Up to this point, I presented a de novo deep model for predicting TF binding affinity from probe sequences. I maintained that integrating temporal features into a model can result in significant improvements. I tested the idea and benchmarked the resulting model as a proof of concept. By using models trained by authors of [34] and comparing their prediction performance with those of our model, I showed that using multiple convolutional kernels can in fact significantly improve the performance. This is contrary to the claim made in [34] that one motif model can represent most TFs up to a negligible small reduction in accuracy, if trained properly, as we have used the same models trained by them and showed the margin is relatively large. While DeeperBind performs well on most of the data we explored in this chapter, in my view, it does not address the followings properly:

1. Because of the way PBMs are designed, any given probe and its reverse complement are effectively considered equal. As a consequence of this, theoretically any trained model will relearn the reverse complement of a kernel that represents a motif. This means that half of the convolutional kernels will be wasted as they can be clearly defined by other kernels. In the next section, I will show that we can mitigate this shortcoming by adding non-trainable weights to the model and thereby improving both model performance and its comprehensiveness while keeping the number of trainable weights fixed.

2. One of the main motivations of utilizing an LSTM network was that it can capture complex relations between the inputs given to it at each time point. Ideally that should translate to capturing higher order positional dependencies. For example, a real binding site followed by certain short (e.g. 3 nt) sites may exhibit a significantly higher binding preference compared to those without such short features. Because of the way that DeeperBind is designed, it may not learn such short motifs. This is because the kernels used by the current architecture have the same shape and the only way to tune the weights of a kernel to fire when such motifs exist, is to have it zero-padded and have only the inner weights trained. Unfortunately, even if such features were powerful enough to impact the loss function in a noticeable way, there would be no way to guarantee finding such kernel weights and therefore, it would be desirable to have this directly encoded into our model. In the next section, I will also address this shortcoming by borrowing from the idea of the Inception network [49, 14, 50] that was introduced recently, by Google Inc.

3. Last but not least, the way we trained the proposed model makes it sensitive to the noise that is available in the measurements. The source for such noises can be traced in batch effects, improper/noisy scanning of the spots, inaccurate normalization of the scanned intensities and the existence of other contributing and confounding fac-

tors that play a role in the binding process (even though this might be less problematic in an in-vitro experiment). Under these conditions, trying to learn a noisy continuous value will not result in a robust prediction model and can affect the generalizability of the trained model. In the following section, I try to address this issue and design and evaluate variants of the DeeperBind that can more effectively deal with the noise in measurements.

2.7 DeeperBind Plus: From Intensities to Relative Preferences

In the previous sections, I proposed DeeperBind, a de novo deep model for prediction of protein binding affinity which was designed specifically for modeling data from in-vitro experiments, most notably protein binding microarrays (PBM). I showed that it surpasses the significant majority of baselines on a number of metrics. DeeperBind uses a number of 1D convolutional kernels of the same sizes to generate useful local features. To pick up positional information we fed CNN feature maps into a two-layer long short-term memory network (LSTM). LSTM works by consolidation of these local features to predict a continuous score that represents the binding affinity of the target TF to the underlying probe.

Despite its remarkable performance our design cannot explicitly address several complexities. Some of these limitations that we will try to address in this section are as follows. First, because of the way PBMs are designed, any given probe and its reverse complement are effectively considered equal. As a consequence, theoretically any trained model will relearn the reverse complement of a kernel that represents a motif. This means that half of the capacity of convolutional kernels will be wasted as they will reflect the same relation. Second, even though the LSTM component of DeeperBind is expected to capture complex relations between the inputs given to it at each time point, it may not be able to learn specific yet important higher order patterns in the sequences. For example, a real binding site followed by certain short (e.g. 3 nt) sites may exhibit a significantly higher binding preferences compared to those without such short features. DeeperBind, the way it is designed, may not properly learn such short motifs. This is because we are trying to learn both features that are distributed more globally (i.e. the binding motifs) and short locally distributed features with kernels of equal size, and the only way to adjust kernel weights for capturing both types is to have some of them zero-padded and fixed around the boundary while the inner weights being trained. Furthermore, such features are often too weak to capture and hence won't impact the loss function in a noticeable way and therefore, it is desirable to have them encoded directly into our models. Finally, the way we trained DeeperBind makes it sensitive to the noise that is available in the measured intensities. This noise could be due to elements such as batch effects, biases, artifacts, improper measurements, inaccurate normalization of the scanned intensities and the existence of other contributing and confounding factors that play a role in the binding process[11]. This calls for a training paradigm that is more robust to such noises.

In the following sections, I revise our previous architecture to address the foregoing complications through integration of additional components into the model as well as utilizing a different training pattern. Lastly, from all modifications considered in this section, we will combine all the effective ones into a single framework that is end-to-end trainable and can be used by researchers for analysis of patterns of regulatory binding. Because our focus here is to improve the performance of the previously introduced model, and since that model was shown to outperform all other baselines, we will henceforth limit our analysis comparisons to the set of DeeperBind and its variants.

2.7.1 Using Non-Trainable Convolution Kernels Obviates the Need for Learning Redundant Patterns

As discussed in chapter 1, PBM works by adding a tagged protein to microarray spots that include double stranded synthesized DNA sequences. The binding preference is then estimated according to the number of probe replicates in each spot as well as the level of

[11]Although this factor is less problematic in in-vitro experiments but still do exist.

intensity due to the binding reaction between the TF and the probe replicates. This binding can take place on any of the two strands of the probe. In fact, both can contribute to the overall signal intensity. Hence, the approach adopted in the previous chapter would lead to training redundant convolutional kernels to account for the motif as well as its reverse complement. This has two implications, first, we are wasting the capacity of the model by training redundant weights, and second, the effective size of the training set will be reduced which makes the model more prone to overfitting.

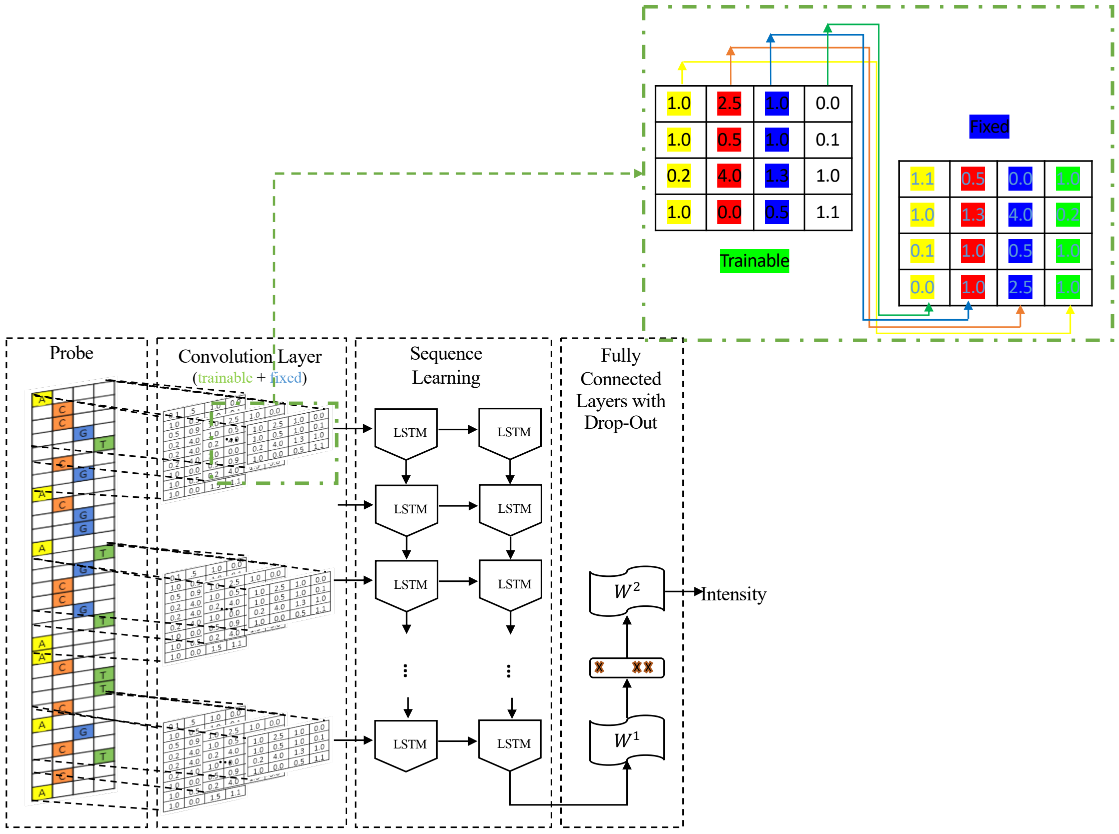


Figure 2.7: Using non-trainable weights, we can capture the reverse complementarity in binding more efficiently.

To handle identification of motifs on both strands properly, we cloned the CNN kernels into non-trainable tensors and then transposed them along the first two dimensions (i.e. position and channel[12], see Figure 2.7 for illustration). Moreover, we used separate

[12]This assumes that the A, C, G, and T nucleotides are in order, encoded with numbers 0,1,2, and 3.

bias weights for the replicate filters and concatenated the resulting feature maps from both trainable and non-trainable kernels, on the depth axis, and subsequently fed the resulting tensor into the LSTM network. Furthermore, we found out that using Leaky Rectified Linear Units [51] (LReLU) results in a slightly faster training trajectory and improves the performance a bit. As a results, in all following models I will be replacing ReLUs with LReLUs. Furthermore, the initial learning rates for all future models were empirically selected to achieve the best convergence trajectory. Aside from the aforementioned changes, no hyper-parameter (e.g. number of trainable kernels, number of replicate models, etc.) was altered for the model implemented in this section.

Table 2.7 compares the rank correlation between the predicted normalized intensities and the measured ones, both for DeeperBind and DeeperBind^{rc} and computed for both sets of probes (i.e. positive vs. all). Interestingly, the addition of non-trainable weights into the model has given rise to consistent improvements for all transcription factors except the first array of Rap1 which can be attributed to the noise in the underlying assay and possibly unaccounted confounding factors. Another interesting observation is that as opposed to the first array in Rap1, in case of the second array, the new model appears to be generalizing well to the positive probes set, and as a result, significantly outperforms the former model. Not only that, the model is also doing better on the set of all probes. These observations suggest that by modeling the binding reaction in a way that is consistent with the technology, we can learn more biologically sound models and reduce the impact of overfitting, despite a slight increase in model complexity[13].

2.7.2 Using Batch Normalization

The observations we made in Section 2.7.1 suggests that despite notable performance improvements, some level of overfitting is still manifested in the predictions on the test sets. This is especially the case for the Rap1 TF suggesting the need for an effective regular-

[13]Note that even though we have not increased the number of convolutional kernels, we did inevitably increase the number of weights in the fully connected and LSTM layers.

Table 2.7: Comparison of the rank correlation metric between DeeperBind and DeeperBind^{rc} for positive and all probes.

TFs	Test Array	Positive Probes		All Probes	
		DeeperBind	DeeperBind^{rc}	DeeperBind	DeeperBind^{rc}
Cbf1	#1	0.64	**0.73**	0.61	**0.69**
	#2	0.72	**0.74**	0.61	**0.63**
CEH-22	#1	0.6	**0.65**	0.54	**0.59**
	#2	0.6	**0.62**	0.45	**0.51**
Rap1	#1	**0.04**	0.03	**0.33**	0.24
	#2	0.08	**0.46**	0.23	**0.28**
Zif268	#1	0.6	**0.61**	**0.59**	0.58
	#2	0.53	**0.57**	0.47	**0.48**
Oct-1	#1	0.46	**0.51**	0.7	**0.71**
	#2	0.47	**0.51**	0.64	**0.65**

ization technique. One efficient and successfully tested recent regularization technique is Batch Normalization (BN) [15]. One obstacle to efficient training is that when a new batch of data is provided to the model for training, the input data distribution changes. This change in distribution (aka internal covariate shift,) pushes layer weights in a wrong direction to account for the change in batch data distribution, hindering the overall training process. At minimum, we would like to have our input data batches to be mean and variance-invariant, which can be performed by the BN technique. By adjusting the mean and variance of the inputs to each layer BN enhances training of each layer in a more independent and consistent manner, and as a result, allows the use of higher learning rates leading to faster convergence.

Table 2.8 compares the rank correlation metric between the new model (DeeperBind^{rc-bn}) and the one trained in the previous section. From this table, we can observe that while the computed performances remain about the same, for the positive set of the first array of Rap1 (i.e. the only array for which the previous model was beaten by the classical baselines, and by a large margin), we observe a significant improvement of the rank correlation, confirming our previous suspicion about an overfitting due to the size and possibly noise in

Table 2.8: Comparison of the rank correlation metric between DeeperBind^{rc} and DeeperBind^{rc-bn} for positive and all probes.

TFs	Test Array	Positive Probes		All Probes	
		DPB^{rc}	DPB^{rc-bn}	DPB^{rc}	DPB^{rc-bn}
Cbf1	#1	**0.73**	0.72	**0.69**	0.68
	#2	0.74	**0.76**	0.63	**0.66**
CEH-22	#1	**0.65**	0.64	**0.59**	0.57
	#2	**0.62**	0.61	**0.51**	0.49
Rap1	#1	0.03	**0.22**	0.24	**0.3**
	#2	0.46	**0.46**	0.28	**0.29**
Zif268	#1	**0.61**	0.6	**0.58**	0.57
	#2	**0.57**	0.54	0.48	**0.49**
Oct-1	#1	0.51	**0.51**	**0.71**	0.68
	#2	0.51	**0.52**	0.65	**0.66**

this array.

2.7.3 Inception

Our experiments so far have shown that by the integration of the one-dimensional convolutional neural networks and recurrent neural networks we can efficiently train kernels that inform the existence of motifs that are attractive to the transcription factors. However, as discussed before, traditional convnets cannot efficiently learn the existence of all available features, especially the genomic features that are shorter in length. Such features are prevalent on DNA and can have lengths as short as 1-2 base pairs (e.g. GC content, splicing sites, etc). For example, using CNNs with a single convolutional kernel of size 1 base pair, combined with an LSTM network, we can infer the GC content of a given probe.

The developed models so far, do not lend themselves well to learning such short features. At best, if such short features are very well represented in the data, larger kernels may be able to learn them, though inefficiently [14]. Thus, the solution is to model features

[14]This is because the kernels used by the CNN component of those models have the same shape and the only way to tune them is to adjust the weights pertaining to a few corresponding positions, and setting the remaining ones to (near) zero.

of different lengths directly. This can be done by borrowing from the idea utilized in the [14] by creating convolutional feature maps using kernels of different sizes (see Figure 2.8). As such, I added three kernels of each of the four lengths 1, 3, 5, and 7 and used convolutions with padding[15] and trained the resulting model on the training sets. Table 2.9 lists the correlation rank metric for the new model as well as the model illustrated in the previous section. According to the table, the inception model consistently outperforms the previous one, and often by a large margin, on both sets. Interestingly, it also improves the predictions for the first array of Rap1, suggesting that the the kernel size of 11, was not an ideal size for modeling the binding preference of this protein.

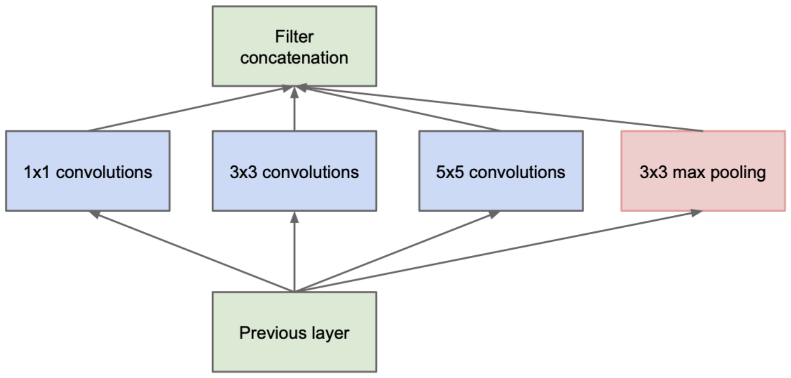


Figure 2.8: A schematic view of an example Inception module. Kernels of different sizes generate feature maps (figure from [14]).

2.7.4 Enhancing the Training Process by Learning Relative Preferences Instead of Normalized Intensities

As discussed before, the measurements performed by the PBM technology is confounded by multiple factors, including the measurement noise and batch effects (c.f. Section 2.6). This is also evident by the sub-optimal reproducibility of the measurements in replicate arrays. Under these conditions, trying to learn a noisy continuous value may not result in a robust predictive model and can affect its generalizability. An alternative to training a

[15]To generate feature maps of same sizes, from each type of kernel.

Table 2.9: Comparison of the rank correlation metric between DeeperBind^{rc-bn} and DeeperBind$^{rc-bn-inc}$ for positive and all probes.

TFs	Test Array	Positive Probes		All Probes	
		DPB^{rc-bn}	DPB$^{rc-bn-inc}$	DPB^{rc-bn}	DPB$^{rc-bn-inc}$
Cbf1	#1	0.72	**0.77**	0.68	**0.75**
	#2	0.76	**0.79**	0.66	**0.71**
CEH-22	#1	0.64	**0.66**	0.57	**0.6**
	#2	0.61	**0.65**	0.49	**0.54**
Rap1	#1	0.22	**0.28**	0.3	**0.37**
	#2	0.46	**0.46**	0.29	**0.38**
Zif268	#1	0.6	**0.63**	**0.57**	0.56
	#2	0.54	**0.61**	0.49	**0.5**
Oct-1	#1	0.51	**0.56**	0.68	**0.7**
	#2	0.52	**0.57**	0.66	**0.68**

regression model is to turn it into a classification problem that learns to compare pairs of inputs by their binding preference. To do that, I cloned DeeperBind^{rc-bn} into two sibling models and tied their weights together. This means that in the new model, we are not adding any new weights, hence little computational burden is introduced. Each sibling model in the overall architecture receives one probe and generates one continuous value. The two scores are then concatenated together and fed into a softmax activation layer (which again does not add any new parameter to the overall model) and trained using the cross-entropy loss. Altogether, the final model learns a slightly different objective, which is the relative preference of probes. Overall, the weights are now learned such that given any pair of probes, the model can predict which one is more favored by the TF[16]. Finally, we can treat the continuous output from each sibling model as a score for its binding preference against the TF and use it for computing the correlation rank metric. Such a model architecture is a reduced version of the so called Siamese network [37] (c.f. Section 3.1.1).

To train this model, I generated random pairs of probes whose measured normalized intensities were different by a predefined margin, and trained it while leaving other hyper-

[16]By our assumption no two probes can be equally attracted by the TF.

Table 2.10: Comparison of the rank correlation metric between DeeperBind^{rc-bn} and DeeperBind$^{rc-bn-sia}$ for positive and all probes.

TFs	Test Array	Positive Probes		All Probes	
		DPB^{rc-bn}	DPB$^{rc-bn-sia}$	DPB^{rc-bn}	DPB$^{rc-bn-sia}$
Cbf1	#1	0.72	**0.72**	0.68	**0.81**
	#2	**0.76**	0.72	0.66	**0.77**
CEH-22	#1	**0.64**	0.59	0.57	**0.61**
	#2	**0.61**	0.58	0.49	**0.58**
Rap1	#1	**0.22**	-0.1	0.3	**0.42**
	#2	**0.46**	0.29	0.29	**0.42**
Zif268	#1	**0.6**	0.59	**0.57**	0.56
	#2	**0.54**	0.49	0.49	**0.5**
Oct-1	#1	**0.51**	0.41	0.68	**0.72**
	#2	**0.52**	0.49	0.66	**0.69**

parameters intact. Table 2.10 compares the correlation rank metric between the new model and DeeperBind^{rc-bn}. Interestingly, while on the set of all probes, we get a significant boost in performance, this is the opposite for the set of positive probes. This is because when randomly choosing input pairs, we did not introduce any bias in the selection process, and hence, the input pairs are more likely to be chosen from the much bigger set of negative probes. That way, we achieve a more performant model but at cost of less accurate predictions on the small set of positive probes. In other words, if our goal was to train a model that works best for the positive probes, we could generate training sets from that set and observe similar improvements[17]. Furthermore, given that the model proposed here is the same as the previous model, DeeperBind^{rc-bn}, but trained differently, we conclude that there are better configuration of weights in the very high dimensional non-convex space of parameters, that are more biologically sound. In other words, in order to fully harness the power of such models, choosing the right training strategy matters and have a huge impact on the performance of the final model. As a result, we can use the proposed training strategy to direct the training trajectory towards better local optima[18].

[17]Which is what many traditional models such as [33] do.

[18]In Chapters 3 and 3 I leverage the same technique to achieve state of the art performance in other domains.

Table 2.11: Comparison of the rank correlation metric between DeeperBind$^{rc-bn-inc}$ and DeeperBind$^{rc-bn-sia}$ for positive and all probes (superscripts $^{rc-bn}$ are dropped for brevity).

TFs	Test Array	Positive Probes		All Probes	
		DPB^{inc}	DPB^{sia}	DPB^{inc}	DPB^{sia}
Cbf1	#1	**0.77**	0.72	0.75	**0.81**
	#2	**0.79**	0.72	0.71	**0.77**
CEH-22	#1	**0.66**	0.59	0.6	**0.61**
	#2	**0.65**	0.58	0.54	**0.58**
Rap1	#1	**0.28**	-0.1	0.37	**0.42**
	#2	**0.46**	0.29	0.38	**0.42**
Zif268	#1	**0.63**	0.59	0.56	**0.56**
	#2	**0.61**	0.49	0.5	**0.5**
Oct-1	#1	**0.56**	0.41	0.7	**0.72**
	#2	**0.57**	0.49	0.68	**0.69**

To see how DeeperBind$^{rc-bn-sai}$ model is doing compared to the previous successful model, DeeperBind$^{rc-bn-inc}$, on all arrays, I listed the rank metric for both, in Table 2.11. Interestingly, according to the table, training strategy appears to play a more important role than using inception modules which itself was the most successful adjustment to the original DeeperBind model. This raises the question of whether or not integrating both techniques results in a synergistic boost in the performance.

2.7.5 Training an Inception-Based DeeperBind Using the Siamese-Based Training Strategy

In the previous sections, I showed that just by changing the underlying task from prediction of normalized signal intensities to the prediction of relative preferences of pairs of probes, and without altering the core model, we can achieve significant improvements. We also observed that by directly modeling genomic features of different lengths we can train more accurate models. In this section, I explore whether the hybrid of the two yields any synergy. Thus, I developed DeeperBind$^{rc-bn-sia-inc}$ and trained it on the set of arrays as in the previous sections and evaluated its performance, as listed in Tables 2.12 and 2.13. Interesting, the hybrid model is doing better that both individual model for all arrays when

Table 2.12: Comparison of the rank correlation metric between DeeperBind$^{rc-bn-sia}$ and DeeperBind$^{rc-bn-inc-sia}$ for positive and all probes (superscripts $^{rc-bn}$ are dropped for brevity.)

TFs	Test Array	Positive Probes		All Probes	
		DPB^{sia}	DPB$^{inc-sia}$	DPB^{sia}	DPB$^{inc-sia}$
Cbf1	#1	**0.72**	0.69	0.81	**0.82**
	#2	0.72	**0.74**	0.77	**0.79**
CEH-22	#1	0.59	**0.66**	0.61	**0.64**
	#2	0.58	**0.67**	0.58	**0.62**
Rap1	#1	-0.1	**-0.05**	0.42	**0.43**
	#2	**0.29**	0.25	0.42	**0.47**
Zif268	#1	**0.59**	0.57	0.56	**0.58**
	#2	0.49	**0.5**	0.5	**0.51**
Oct-1	#1	0.41	**0.41**	0.72	**0.72**
	#2	0.49	**0.5**	0.69	**0.71**

all probes are considered. The Spearman rank correlation for the hybrid model is as high as 0.82 for the Cbf transcription factor. Such a high performance is especially remarkable when comparing to the computed correlation between median intensities on the two arrays, as reported in Table 2.1. Note that the correlations reported in that table are computed after training k-mer models on the ground truths (i.e. measured normalized intensities). Therefore, a significant amount of information can not be captured by such models which the hybrid model (henceforth, DeeperBind Plus) can leverage, leading to promising performances. Lastly, depending on the analysis objectives, one can learn DeeperBind Plus on pairs of probes that are randomly sampled from the positive set, and thereby utilizing the same synergistic improvements there.

Table 2.13: Comparison of the rank correlation metric between DeeperBind$^{rc-bn-inc}$ and DeeperBind$^{rc-bn-inc-sia}$ for positive and all probes (superscripts $^{rc-bn}$ are dropped for brevity.)

TFs	Test Array	Positive Probes		All Probes	
		DPB^{inc}	DPB$^{inc-sia}$	DPB^{inc}	DPB$^{inc-sia}$
Cbf1	#1	**0.77**	0.69	0.75	**0.82**
	#2	**0.79**	0.74	0.71	**0.79**
CEH-22	#1	0.66	**0.66**	0.6	**0.64**
	#2	0.65	**0.67**	0.54	**0.62**
Rap1	#1	**0.28**	-0.05	0.37	**0.43**
	#2	**0.46**	0.25	0.38	**0.47**
Zif268	#1	**0.63**	0.57	0.56	**0.58**
	#2	**0.61**	0.5	0.5	**0.51**
Oct-1	#1	**0.56**	0.41	0.7	**0.72**
	#2	**0.57**	0.5	0.68	**0.71**

2.8 Conclusion, Remaining Unsolved Challenges, and Future Work

In this chapter, I developed and benchmarked a series of models for prediction of protein binding preferences, starting from DeeperBind and ending to DeeperBind Plus. Through this chapter, I took advantage of advanced techniques that have been successfully employed by the machine learning community, in other domains, such as vision. While the models in this chapter were trained on protein binding microarrays, they can potentially be applied to other types of data, as well. I showed that in order to fully harness the power of genomic features and the underlying technology, one needs to directly model each aspect by including the right component in the model. Specifically, the aspects that we addressed extensively were, the overall temporal contribution of motifs and genomic features on probes, the potentially varying lengths of genomic features, the reverse complimentarity that is specific to the underlying technology, and the existence of noise in measurements. Furthermore, I showed that the current models that are available for the task of interest in this chapter can yield higher performance rates, if trained properly. I showed that by slightly changing the underlying problem from a regression model predicting normalized intensity into a classifier that learns to compare probes' corresponding binding affinities, we can gain more prediction power from such models.

The work presented in this chapter can be extended from different perspectives. First, while using the recurrent neural networks we hope that sequential dependencies are efficiently captured, studies have shown that we can do better by attending to the whole sequence [52]. Through attention the hope is to learn patterns distant from each other. Another unexplored dimension is the inclusion of other external information for further improvement of binding preference, and of course, in more realistic settings. One of the most fascinating and yet challenging directions is the incorporation of protein information (e.g. protein's 3D structure, within and between family conservation, etc.)) in the form of additional features. Furthermore, we can include genomic structural information into the model, all of which help accounting for other layers complexities that are present, and as discussed before. While the family of DeeperBind models, can address some of these challenges indirectly, yet as we extensively discussed in this chapter, directly modeling such complexities can be a more effective approach. Furthermore, there are yet other complexities that that out models fall short to address, even indirectly, such as in-vivo considerations, all of which can be accounted for in future work.

CHAPTER 3
FROM REGRESSION TO CLASSIFICATION: UNRAVELING HIDDEN STRUCTURES IN GENOME WIDE ASSOCIATION STUDIES WHEN DATA SCARCITY IS A CHALLENGE

3.1 Introduction

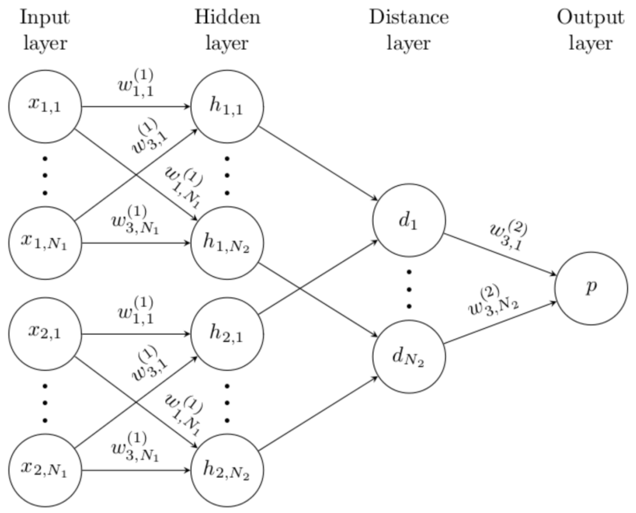
Two key challenges when solving problems in Bioinformatics-related domains (such as GWA studies,) are the data scarcity and existence of noisy/incomplete features/targets. Often times, acquiring enough data or labeled data, for that matter, is costly or time consuming. This renders a comprehensive data acquisition an infeasible task. The noise/uncertainty that is often present in data can make the prediction tasks with limited data even more challenging. To get around the challenges posed by these two, the mainstream strategy is to reduce model complexity. While this turns out to be an effective strategy (although at cost of adding bias to the model), more can be done to benefit from small noisy datasets with continuous targets. In this chapter, we will utilize a more effective approach inspired from Siamese networks, which is a popular technique in the image processing domain, and show that we can learn powerful models by learning relations instead of continuous target values, and thereby, creating a relationship graph that quantifies similarities between pairs of samples. We apply this strategy to two important problems in the GWA domain (one in this chapter and one in chapter 4) and share new insights about the outcome.

The rest of this chapter is divided into three main parts. In the first part, we introduce Siamese nets and the idea behind these models. Then we discuss our proposed model, eQTLNet, in section 3.2. We will show that by learning pairwise relationships, we can train a parametric model for prediction of Expression Quantitative Trait Loci (eQTLs) with a performance better or in par, with the state-of-the-art models. We will then evaluate eQTLNet and compare and contrast its performance with two baseline models, one state-of-the-art model, and one which is the result of a reduction of our model to the classical regression framework. We will argue that parametric approaches can be superior to the non-parametric ones, if trained properly. In Chapter 4, we will revisit another application of training regression models in the context of learning relations but for a different problem where we will share interesting insights and findings for the first time.

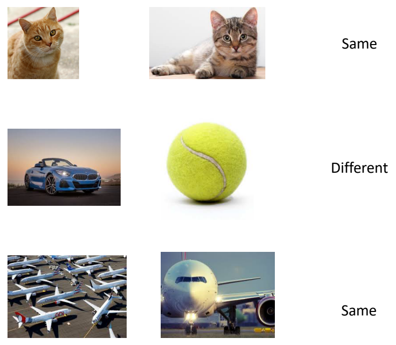
3.1.1 Siamese Networks

Learning good features poses a challenge when little supervised data is available [38]. Siamese networks [37] was first proposed in 1993 for the signature verification task. Figure 3.1 depicts the architecture of a Siamese network. A Siamese net is a network comprised of twin networks with shared weights. Each wing in a Siamese net accepts a different input of either the same or different type[1]. At the highest level, the distance between the representations of each input is computed which serves as a measure for the final classification task. The weight tying mechanism guarantees that two very similar inputs are not mapped to different points in the latent space that the network maps to. Furthermore, the networks are symmetric in the sense that if two input data are switched between the the twin networks, the distance remain the same (assuming that the two inputs are of the same type). Different distance metrics may be used depending on the prediction task and the data modality. The most popular distance layers are contrastive energy [53] , L1 (or L2) weighted distances [54], and the dot product between the two representations.

[1]In case of different input data types, the weights for twin networks are not necessarily tied



(a) A 2-hidden layer Siamese net for classification along



(b) Siamese networks are trained to classify pairs of input samples.

Figure 3.1: Siamese networks use two sub-networks (often) with tied parameters and a layer to compute the distance between the representation generated by each sub-network. This distance informs the overall network of the similarity between the input pair. Figures from [37].

3.2 eQTLNet: Enhancing the Prediction of Expression Quantitative Trait Loci (eQTLs) with Siamese Networks

Genome-wide association studies have led to identification of millions of genetic variants and thousands of loci hosting them. These data have been shown to be linked with numerous complex diseases, though in an involved way. During the past decade, genotype data have been enormously used for genetic prediction of complex traits and gene expression levels. GWA studies have unraveled an unprecedented level of insights into the effects of variants on phenotypes of interest. Successful and accurate prediction of phenotypes have important implications in different domains such as personalized medicine in human and genomic selection in animal and plant breeding [55, 56]. This however comes with key challenges that restrict data driven methodologies. Among the key challenges is the complex polygenic relationships between the variants. Studies have shown that majority of complex traits and common diseases have a polygenic structure [55, 57, 58, 59, 60] and are partially due to hundreds to thousands of SNPs discovered or to be discovered, each with a small contribution. As a result, models that integrate the effect of a few variants can only explain a proportion of phenotypic variances and therefore are not suitable for predicting phenotypes with polygenic architecture. The high-dimensionality of phenotype-linked variants necessitates the existence of datasets larger than what we have today, by at least orders magnitude, in order to be able to train relatively complex models. Even if such datasets existed still a significant portion of phenotypic variabilities is only explained by non-genetic factors (i.e. environment). Due to the foregoing challenges, all state-of-the-art approaches today, choose to reduce model complexity to avoid getting a model with high variance (i.e. to avoid overfitting).

Prediction of continuous targets in GWAS is often a more challenging task compared to the prediction of discrete end-points (e.g. diseased vs normal) as regression models suffer more severely from adverse effects of uncertainty and noise in data as well as phenotype values. To the best of my knowledge, the most successful GWA models are based on regression techniques. These models are often Bayesian in nature, i.e. they assume some prior on the effect sizes of variants and optimize for the posterior. Most of these methods are parametric, while more recently non-parametric approaches have offered promises and resulted in higher performance compared to the parametric ones [55]. The non-parametric methods are different from the parametric ones in that no ad-hoc assumption on the effect-size distribution is assumed a priori. In fact, the prior distribution of effect sizes for a trait or disease can depend on several factors such as the disease itself, the individual SNPs effect sizes, the number of causal variants, and their minor allele frequencies (MAFs) [55]. Therefore, it makes more sense to tailor a prior for effect sizes of each dataset to achieve a more accurate model. Recently, latent Dirichlet Process Regression (DPR) method [55] has been introduced and has been shown to outperform its predecessor models. As suggested by the name DPR uses the Dirichlet process to infer an effect size distribution for the SNPs at hand. Being a non-parametric distribution, would mean an infinite degree of freedom for the distribution to be learned and hence, effectively, DPR can learn any prior distribution if enough data is provided.

The focus of this section is to explore a new way of training models for prediction of Expression Quantitative Trait Loci (eQTLs) from cis-SNPs. eQTLs are loci that explain, often partially, the variation in expression levels of mRNAs [61]. The fact that majority of associated variants are in regulatory non-coding regions of the DNA, suggests that the functional mechanisms involved in the disease onset is mostly through the regulation of gene expression. Prediction of eQTLs is different from other complex traits in that the phenotype is typically a product of only loci. eQTLs can be cis or trans. The latter is different from the cis in that variants are located on a distant location, often on a different chromosome. Most of the eQTLs found so far are of cis type [62]. Likewise, in this study, we consider variants that are located in the vicinity of the underlying genes. Due to its proven state-of-the-art performance, we selected DPR as a baseline for benchmarking our proposed method. We evaluated the performance of a regression model that is reduced from our proposed model to shed more light on the extra performance gain that is due to the difference in the new training strategy, i.e. learning pairwise relations instead of the expression level. We tested our model both on simulated data and real-world datasets discussed bellow, and analyzed the outcome.

3.3 Datasets

For evaluation and benchmarking, we apply our model along with the baselines to two real-world datasets, detailed bellow. We also tried these methods on a series of synthetically generated datasets, derived from the CAGE blood dataset to show how successful each method is in explaining phenotypic variances.

3.3.1 Predictors of Response to Standardized Paediatric Colitis Therapy (PROTECT) Dataset

We retrieved the data from a recently published study [63] conducted to predict the disease course based on clinical activity and initial treatment responses as well as the rectal gene expressions acquired with RNA sequencing before treatment. Samples where pediatric patients aged between 4-17 years who were newly diagnosed with ulcerative colitis in 29 centers located in USA and Canada [64]. The hypothesis was that with the help of clinical features and biological markers we can explain for the variability in the treatment response. For the purpose of our own study, we selected 11266 loci harboring genes along with their 1Mb flanking regions and retrieved variants located in those regions, representing the cis SNPs.

3.3.2 The Consortium for the Architecture of Gene Expression Dataset

The Consortium for the Architecture of Gene Expression (CAGE) is an initiative that accumulates genotype and gene expression of peripheral blood for seven distinct cohorts [65].

Data have been undergone quality control tests (before and after imputation), normalized and imputed to the 1000Genomes project, Phase 1, reference panel. The latest CAGE dataset [66] hosts phenotypes and genotypes for 2765 samples. To avoid the complications due to the population structure, in this study we focused on a single cohort (i.e. the EG-CUT) with 1065 samples. We used the same subset of genes and flanking regions as in the PROTECT dataset and created a dataset of variants and phenotypes for our benchmarking purpose.

3.3.3 Simulated Dataset

While real data helps conducting an unbiased evaluation of our proposed model, synthetic data can be more useful in a controlled analysis of the model, as in the former case, the number of causal variants, their effect sizes, and the relations thereof, are largely unknown. As a result, it is hard to make an accurate estimation of the heritability and the power of the model under scrutiny without having a gold standard. Therefore, besides the PROTECT and the CAGE datasets, we also assayed our model by generating different synthetic datasets derived from the CAGE dataset, through random selection of causal variants and their effect sizes. Specifically, we generated datasets with 10, 100, 500, and 1000 causal SNPs randomly selected from all variants of the loci of interest and randomly assigned effect sizes to each of them, separately for each number of allele counts (i.e. 0, 1, and 2).

3.4 Model Architecture

Current GWA models are regression models that approximate the phenotype values directly from the variants. As discussed in the introduction, such models are often less robust to noise and uncertainty. The idea behind eQTLNet is that instead of learning the phenotype, we can learn relations between them. In other words, let $S = (s_i | i \in 1, \ldots, N)$ be a strict partial order [67] of phenotypes pertaining to N samples such that $s1 > s2 > \cdots >$

$s_{N-1} > s_N$. We can represent this series with a complete[2] directed graph data structure (henceforth, the phenotype graph). Let $G(V, E)$ be a graph with a vertex set V that has a one-to-one correspondence to the samples and E be the set of edges such that if $e = (v_i, v_j) \in E$ iff $s_i > s_j$. The following theorems and corollaries hold for this graph.

Theorem 3.4.1. *G is a Directed Acyclic Graph (DAG).*

Proof. Let's assume by contradiction that a cycle exists in graph G. Let $c = (v_{l_1}, v_{l_2}, \ldots, v_{l_n}, v_{l_1}$ be one such cycle. By construction, an edge e_{ij} is part of the cycle c only if $s_i \geq s_j$ which implies $s_{l_1} \geq s_{l_2} \cdots \geq s_{l_n} \geq s_{l_1}$ and which can't be true as by our assumption, the phenotype values are distinct. □

Corollary 3.4.1.1. *Let* $e = (v_i, v_j) \in E$*, if and only if* $deg_o(v_i) > deg_o v_j$.

Proof. The forward direction follows since for any v_k such that $(v_j, v_k) \in E$ we must have (v_i, v_k) or there will be a cycle in G otherwise. Therefore, $deg_o(v_i)$ must be at least $deg_o(v_j)+1$ where the additional edge is due to $e = (v_i, v_j)$. Likewise, $(v_i, v_j) \notin E$ implies that $(v_j, v_i) \in E$ and therefore, by the same argument, $deg_o(v_i) < deg_o v_j$. □

Theorem 3.4.2. *No two vertices in graph G have similar out degrees.*

Proof. Let's by contradiction assume that there exists two nodes v_i, v_j in G with the same out degrees. Let's assume $s_i \geq s_j$. Hence, there exists an edge from v_i to v_j. Since the number of out-degree edges from v_i is equal to that of v_j, there will be at least one vertex v_k where $e = (v_j, v_k) \in E$ and $(v_i, v_k) \notin E$. Observe that we found a cycle $c = (v_i, v_j, v_k, v_i)$ which is a contradiction as per the previous theorem. □

Corollary 3.4.2.1. *The out degree of vertices in G will be distinct numbers between* 0 *and* $N - 1$.

Corollary 3.4.2.2. *We can reconstruct a strict partial order from G via sorting the vertices by their out degrees in descending order, in* $O(N)$ *time.*

[2]We define a complete directed graph of N vertices to be a graph where the sum of in degree and out degree for each node be equal to $N - 1$.

Proof. The proof follows from Corollary 3.4.1.1. That is, for any two vertices v_i, v_j where $deg_o(v_i) > deg_o v_j$ we have $e = (v_i, v_j) \in E$ and hence $s_i > s_j$. Moreover, we can do the sorting in linear time since by Corollary 3.4.2.1 the out degrees are distinct integers in the range $[0, N-1]$. Note however, that finding the out degree of nodes take $O(E) = O(N^2)$ time, and hence, the overall computational complexity is $O(N^2)$. □

Corollary 3.4.2.3. *G uniquely specifies S, and vice versa.*

Proof. For the proof for the forward statement, observe that by the construction methodology in the proof of Corollary 3.4.2.1, there will be at least one strict partial order S^1 given G. Now, by contradiction, let's assume that there are more than one such orderings. Pick one and call it S^2. Since S^1 is different from S^2, there must be at least one pair of vertices, say v_i, v_j such that $s_i^1 > s_j^1$ and $s_i^2 < s_j^2$. Therefore, by way of construction, both (v_i, v_j) and (v_j, v_i) must belong to the set E, which is not possible. The reverse direction follows directly from the construction of G. □

3.4.1 Inferring Phenotype Graph from Variants Data

Corollary 3.4.2.3 motivates learning a neural network that can ideally represent a complete DAG given the samples (i.e. variants) as it guarantees the uniqueness of the phenotypic partial order of interest. In other words, the goal is to predict the direction of the edge that exists between any pair of samples. Figure 3.2 depicts the architecture of eQTLNet. Inspired by the Siamese models, eQTLNet takes as input the variants from two distinct samples and try to learn a distance metric from which the aforementioned edge direction can be inferred. The network is comprised of two hidden layers. The first layer generates a representation in a latent space, followed by a subtraction layer. The hope is that through training this network, a mapping between the variant space and the latent space is learnt such that the subtracted representation highlights the important differences that impact the phenotype relationship. The result of this difference is then fed into the second hidden layer that generates logit values for a binary classification. We used 5 neurons for the first hidden layer and 2 neurons for the logits due to the two classes being learned. The weights of the sub-networks in the first layer, as well as the weights of each neuron in the logit layer are tied together. We used L_1-norm and L_2-norm regularizers with respective penalty coefficients of 0.01 and 0.002. The strong L_1 coefficient enforces most of the weights to be set to zero during training. This plays the role of a feature reduction stage as majority of the variants are not causal with respect to the phenotype of interest. The L_2-norm regularizer also helps overcoming overfitting due to the scarcity of data. Furthermore, we found out that linear activation results in the best performance. In that case, this model is effectively limiting the search space through a restriction bias [68] and hence can be considered as a matrix factorization method where the first matrix maps the input pairs into a latent space with good a separability between forward and backward edges. For the baselines, we use Dirichlet Process Regression (DPR) as well as the elastic net [69]. Being a parametric non-Bayesian method, the latter can be considered as a reduced version of eQTLNet, but in the regression domain, and thereby can provide us with insight into the utility of the Siamese base learning pattern.

3.4.2 Predicting Phenotypic Ranks on the Test Dataset

As stated by Corollary 3.4.2.2, given a graph (i.e. variants for all pairs of samples) with the corresponding out degrees, we can infer the ranks of each test sample in linear time. This however, holds under a strong assumption that our model can correctly infer the phenotype graph from the test samples. In practice, this assumption often fails to hold and our prediction of edge directions will be noisy. This is due to multiple sources of noises in the data, the missing heritability, as well as the complex relationship between variants and the phenotypes. Therefore, once we infer the phenotype graph from the test sample variants, we need to find a strict partial order that is the most consistent with the underlying graph. We can assign a weight to each edge based on the normalized score that our network predicts for that edge. In other words, the problem now can be stated as follows;

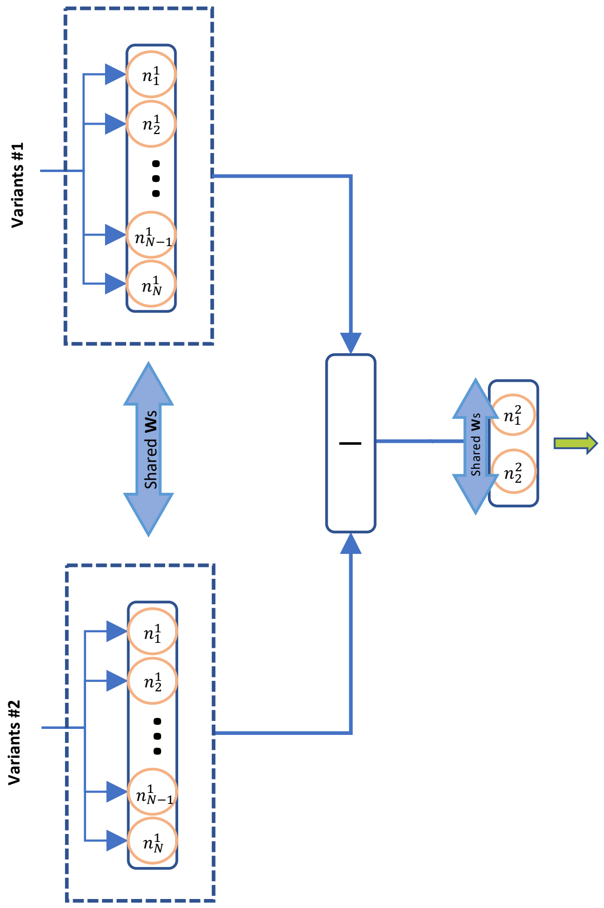


Figure 3.2: eQTLNet's Model architecture.

Let $G(V, E)$ be a noisy directed inferred phenotype graph on a genotypic dataset where for each pair of samples v_i and v_j, either (v_i, v_j) or (v_j, v_i) belongs to E. Find a new directed acyclic graph $G'(V, E')$ on the same set of vertices by selecting some of the edges in E and flipping their direction, such that the given cost function is optimized. Here I consider two cost functions, with the second having the first as a special case, and show that despite the existence of a linear solution for the noiseless case, the solution to both problems are in fact NP complete[3]. We then propose two heuristic approaches for the general problem and explore the effectiveness of the more successful one in the results section.

[3]For a definition of NP completeness the reader can refer to [70].

Cost #1: Minimizing the number of edge flipping.

The objective here is to correct a noisy phenotype graph by flipping the minimum number of edges. Figure 3.3a illustrates one such conversion for a simple example with 4 samples. For the depicted example, the optimal conversion can be achieved by flipping a single edge which results in the strict partial order of $s_2 > s_1 > s_3 > s_4$. We prove that this problem is equivalent to the minimum feedback arc problem, an NP-complete problem, and hence has no polynomial time solution. In the minimum feedback arc set problem the objective is to convert a directed graph into a DAG by removing (instead of reversing) the minimum number of edges (i.e. breaking the cycles in an optimal way).

Problem 3.4.2.1. *The minimum feedback arc problem.*

Given a directed graph, $G(V, E)$, the goal is to find a minimum set of edges so that upon removal of those edges, the graph becomes acyclic. Rigorously speaking, the goal is to find a set $E^\star \subseteq E$ such that $G(V, (E \setminus E^\star))$ is acyclic and $\forall e \in E^\star \ni G(V, (E \setminus E^\star) \setminus e)$ has a cycle.

The minimum feedback arc problem has been shown to be NP complete [71].

Theorem 3.4.3. *Let's $G'(V, E')$ be a noisy phenotype graph inferred by our model. Let's $\overrightarrow{E^\star} \subseteq E'$ be a minimal set of edges that makes $G(V, E' \setminus \overrightarrow{E^\star})$ a DAG and $\overleftarrow{E^\star}$ be the set of edges in $\overrightarrow{E^\star}$ flipped. For any such minimal set $\overrightarrow{E^\star}$, $G(V, E' \setminus \overrightarrow{E^\star})$ is a DAG if and only if $G(V, E' \setminus \overrightarrow{E^\star} \cup \overleftarrow{E^\star})$ is a DAG.*

Proof. The proof for the reverse direction is trivial. For the forward direction, let's assume the right side of the bidirectional implication does not hold, we show that in that case the left side will not hold either.

Let's assume that $G(V, E' \setminus \overrightarrow{E^\star} \cup \overleftarrow{E^\star})$ has a cycle in it. Either this cycle uses all or some of the edges in $\overleftarrow{E^\star}$ or not. If none of the edges in $\overleftarrow{E^\star}$ appear in a cycle, then $G(V, E' \setminus \overrightarrow{E^\star})$ has already a cycle in it, and hence the left side does not hold and we are done. If, on the other hand, some of those edges appear in a cycle in the augmented graph, then let's pick one such edges, $\overleftarrow{e} \in \overleftarrow{E^\star}$, and consider the following two possibilities. First, there exist at least one of the edges in the set $\overrightarrow{E^\star}$ whose inclusion does not lead to a cycle. In which case the minimality of the set $\overrightarrow{E^\star}$ is violated, or second, adding any of those edges in the set will add a cycle to the graph. Let's pick the edge, $\overrightarrow{e} \in \overrightarrow{E^\star}$ which is the flipped version of $\overleftarrow{e}$. Without loss of generality, let's assume that the two cycles intersect only at the source and end vertices of $\overleftarrow{e}$. Let's call these cycles c_1 and c_2, respectively, observe that $(c_1 \setminus \overleftarrow{e}) \cup (c_2 \setminus \overrightarrow{e})$ is a cycle that does not include $\overrightarrow{e}$, and this again violates the minimality of the set $\overrightarrow{E^\star}$. □

Cost #2: Maximizing the sum of edge weights.

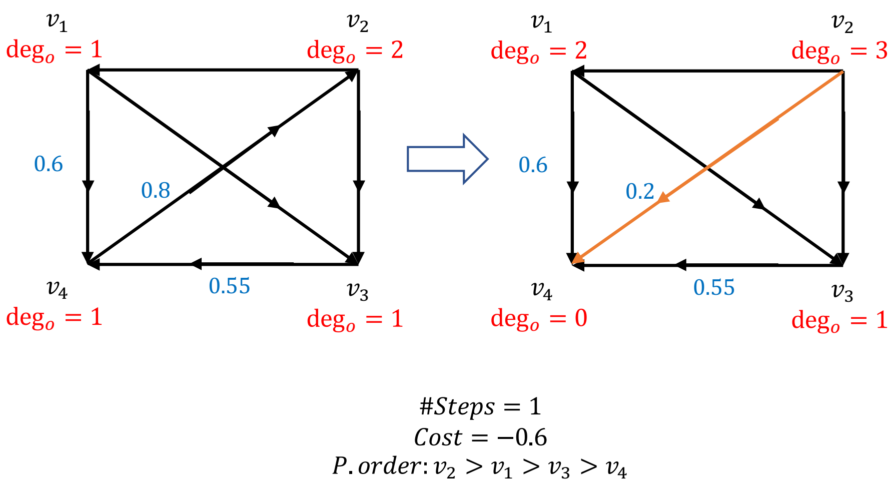
The correction based on the previous cost function does not consider the normalized weights predicted by our model. Figure 3.3a depicts one such example. According to the figure, while the conversion takes place in one step, it is achieved by reversing an edge that is linked with a high predicted probability. This is however, not an appealing reversion. It would make more sense to give higher probability to those edges for which our model predicts with higher certainty. To address this, we can correct the graph in such a way that the sum of predicted weights for the flipped edges is maximized. Figure 3.3b shows one such conversion for the same graph which even though applies more flips, leads to a higher total sum of predicted weights.

While the first cost function often leads to fewer conversion steps, this optimization problem is more favored as it preserves edges that are more likely to be true relations. Once we have a corrected phenotype graph, we can infer sample ranks by sorting the out degrees in linear time. Therefore, the most challenging part is to correct the graph. Unfortunately, this problem, too, is at least as hard as the previous one.

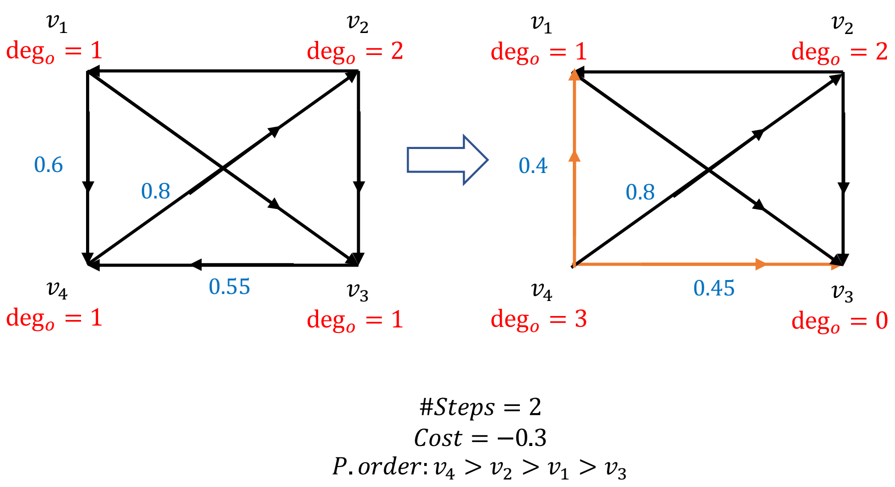
Theorem 3.4.4. *Finding the optimal phenotype graph under cost #2 is at least as hard as the one under cost #1.*

Proof. We prove this theorem by a reduction from the first problem to the second one. Simply assign weights 1 to all edges that belong to the noisy phenotype graph and consequently, weight 0 to the edges in the complement edge set. The conversion is performed in polynomial time, and hence, unless P=NP, if the second problem can be solved in polynomial time, so does the first one. □

As there is no polynomial time solution for finding phenotype graphs, we will resort to



(a) An optimal conversion based on the minimum number of edge flipping.



(b) An optimal conversion based on the edge weights.

Figure 3.3: Correcting a noisy phenotype graph using two different objectives.

heuristics to correct the graph. Bellow, we suggest a greedy solution and a approach based on randomized optimization to find a feasible solution.

1. Integrating Walks of Certain Length.

   For any given edge we can infer the direction by integrating all possible paths from the source to the destination. This can be useful to correct misclassified edges. For example, the predicted score for a given edge (v_i, v_j) might be close to 0.5 and as a result cannot be a reliable prediction. Instead, there might be a series of edges that create a path from v_i to v_j with all their predicted scores being close to 1.0. In such cases, it may be beneficial to integrate those solutions to infer a more accurate score for (v_i, v_j) and hence the right edge direction. This requires us to consider all forward and backward edges between any pair, with their corresponding weights (which sums up to 1.0). One way to integrate all these paths is through successive multiplication of the predicted distance matrix, calculated by applying the softmax function to the computed logits, with its transposed version, to get a distance matrix that considers all paths of length 2. It may be beneficial to consider path of lengths higher than 2 as well, using successive multiplications, and computing a weighted sum up the resulting matrices.

   While this method, can potentially correct some of the mistakes, it does not generate a phenotype graph and the edges can still remain inconsistent and consequently cannot inform an ordering of the phenotypes.

2. Randomized Optimization.

   Another alternative is that instead of trying to correct edge directions individually, we use the original distance matrix to search directly for a ranking of phenotypes that maximizes some objective function this also has the benefit of having a consistent phenotype graph in each iteration. To do that, we can take advantage of randomized optimization techniques, most notably, the Genetic Algorithm [72]. The optimization problem in that case would be very similar to the Traveling Salesman Problem with the cities being mapped to samples and the phenotypic ranks being the the distance in terms of the number cities visited from the first city in the chromosome. Each chromosome was designed to contain an ordered assortment of the ranks of samples. Having an ordered assortment, we define the fitness function of each solution to be the sum of edge weights of the phenotype graph inferred by it. We initialized each chromosome to a random shuffling of integers starting from 0 to $N - 1$. We then used populations of size 100 and evolved them for a 100 generations. To avoid losing good solutions in each population, an elitism policy was adopted where in each generation we passed the individuals with the top 5 fitness values to the next generation. Furthermore, with the mutation probability rate of 0.10 we randomly selected individuals and then two points on them and flipped the order of genes (i.e. ranks) between the two. Finally, for the cross over operator, we selected parents based on a tournament strategy[73] and subsequently chose two points on them, copied all genes enclosed by the two points into the same location in the corresponding offspring, and then filled the remaining positions from the other parent in the same order, but discarding genes that were already copied. Upon termination of the evolution course we selected the fittest chromosome as the final solution.

3.5 eQTLNet Benchmarking Using Simulated and Real Data

3.5.1 Model Parametrization and Dataset Preparation

We trained eQTLNet on the CAGE and PROTECT datasets and on the simulated data generated from the CAGE dataset, as described above. To achieve statistically reliable performance statistics, we randomly selected 250 genes from each of the three datasets, and for each gene, we split the samples into three parts corresponding to the a training, a test, and a held-out set with splitting ratios of 67%, 23%, and 10%, respectively. For the simulated data, we then selected a subset of the variants as causal SNPs that explain the heritability. We repeated the experiments based on different combinations of the number of causal SNPs and the standard deviation of the random residual term that explains for confounding factors. The residual term is drawn from a zero-mean Gaussian distribution with different standard deviations. The causal genotypes were drawn from a uniform random distribution and the phenotype of each individual was calculated as a function of its genotypes, the randomly assigned effect sizes, and the residual term.

To achieve consistent results, the Elastic Net and eQTLNet models were both implemented using the same open source library Python Tensorflow [41] from Google and trained with the stochastic gradient descent optimizer and batch sizes of 250 and 50, respectively. The learning rates were opted using a partial grid search strategy that resulted in the best performance. For DPR, we used the same parameters recommended by its developers.

3.5.2 Simulated Dataset

Scenario #1: Phenotypes Fully Explainable by Genotypes

In our first set of experiments, we we generated four synthetic datasets by setting the number of causal variants to 10, 100, 500, and 1000, respectively. We drew random genotypes from a uniform distribution for the selected SNPs. Furthermore, for each causal SNP and genotype, we randomly assigned an effect size from the range $(0, \frac{1}{\#Causal}]$. Figure 3.4 plots distributions of phenotypes for a gene in each of the above settings. As suggested by the figure, the higher the number of causal variants, the more Gaussian the distribution will appear to be which is consistent with the central limit theorem.

For Elastic Net and DPR methods, we used the predicted phenotypes on the test sets, to construct the corresponding phenotype graphs. We then created the ground truth phenotype graphs from the target phenotypes. Having the graphs from eQTLNet and our baselines, we computed the accuracy of each in classifying the edge directions (henceforth, the predicted graph accuracy). Figure 3.5 depicts the scatter plots of graph accuracies for eQTLNet vs

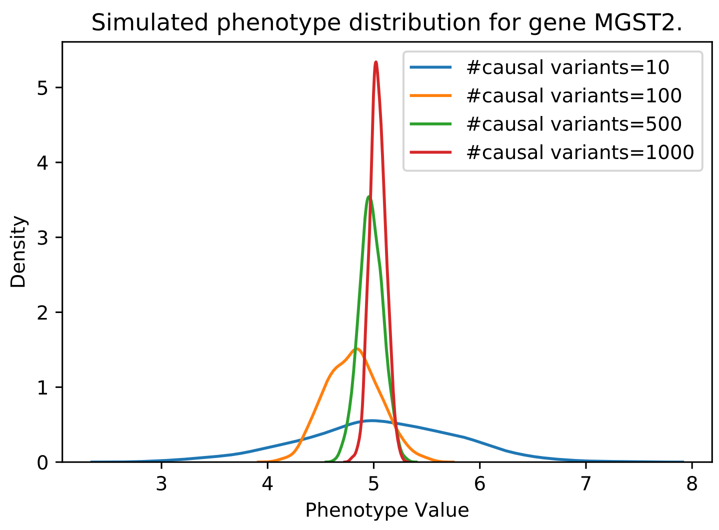


Figure 3.4: Distribution of simulated phenotypes under different simulation settings.

DPR for each experiment when no noise is injected into the predefined association.

A few key observations can be made according to these figures. First, while eQTLNet is doing consistently better, the margin is significantly higher when the number of causal variants is within the range of a couple of hundred SNPs. Specifically, for the case with only 10 causal variants, the data appear to fall into two separate clusters. In the larger cluster DPR is doing relatively similar (in fact slightly better), while in the smaller cluster, eQTL is clearly the winner model, leading to an overall mean accuracy improvement of 1.26%. This could be due to not performing any feature reduction before training a model. A stronger L_1-norm regularizer can also help, however, at cost of lower performance for cases with more causal variants. Another possibility is to integrate multiple models into an ensemble model so that both ends of the spectrum will be accounted for. Second, eQTLNet significantly surpasses DPR with regard to the graph accuracy, when the number of causal variants is around a couple of hundred variants. This improvement while still exits, drops noticeably for loci with about 1000 causal variants. Third, for the last two experiments, the associations for a few genes were captured by both methods with a relatively high accuracy. This could be due to a random assignment of effect sizes to the genotypes that has

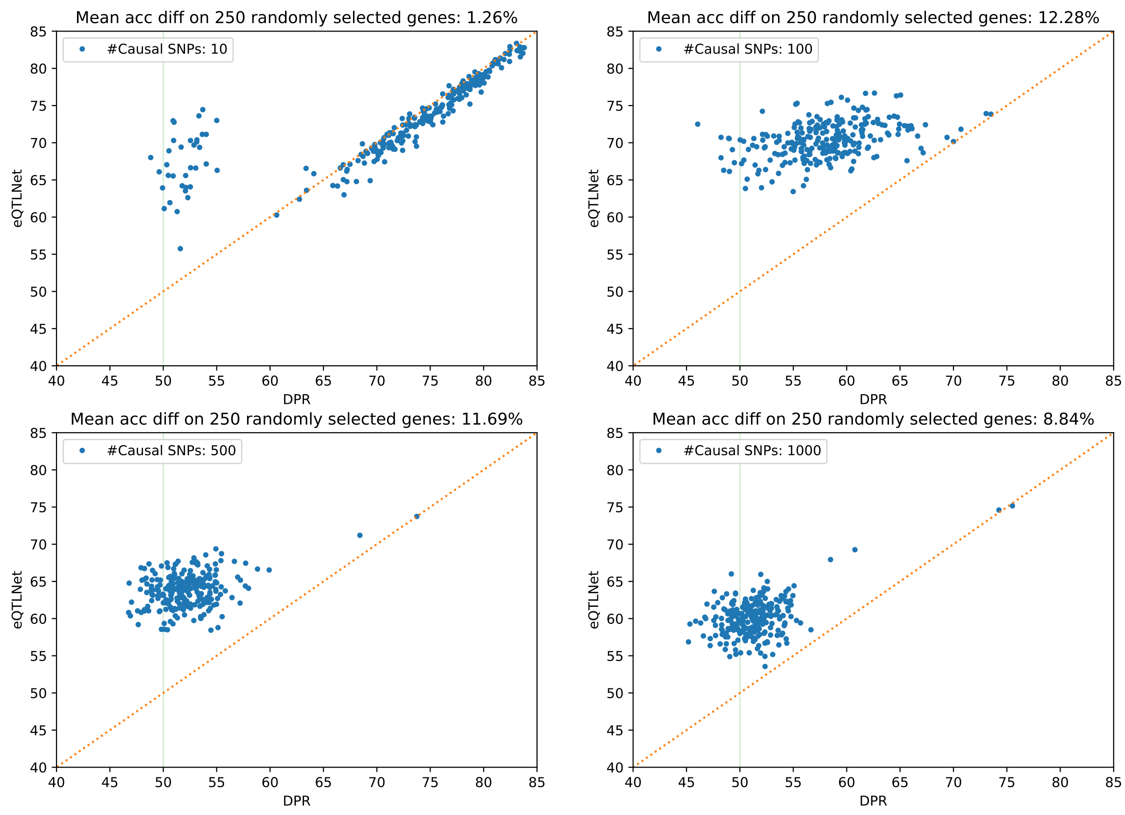


Figure 3.5: Scatter plot of graph accuracies for different number of causal SNPs, in the absence of any residual noise. The mean improvement in is note in the title of each subfigure.

led to a linear association that could be more easily discovered by both methods. Fourth, as the number of causal variants increase, DPR tends to behave as a random classifier for many samples. In fact, for the case with 1000 causal variants, for majority of the genes DPR classifies directions with a classification accuracy of less than 50%. Finally, trained eQTLNet models show a lower variance[4]. Figure 3.6 shows the distribution of graph accuracies for each method. The lower variance is clearly observed for the first two experiments. The difference in variance appears to diminish as the number of causal variants increase, although the difference in prediction accuracy still remains significant.

The next question we addressed was to see how the improvement in accuracy is linked to the training pattern itself. To do that, we reduced the eQTLNet model to an elastic net

[4]See the discussion of bias-variance relation in chapter 4.

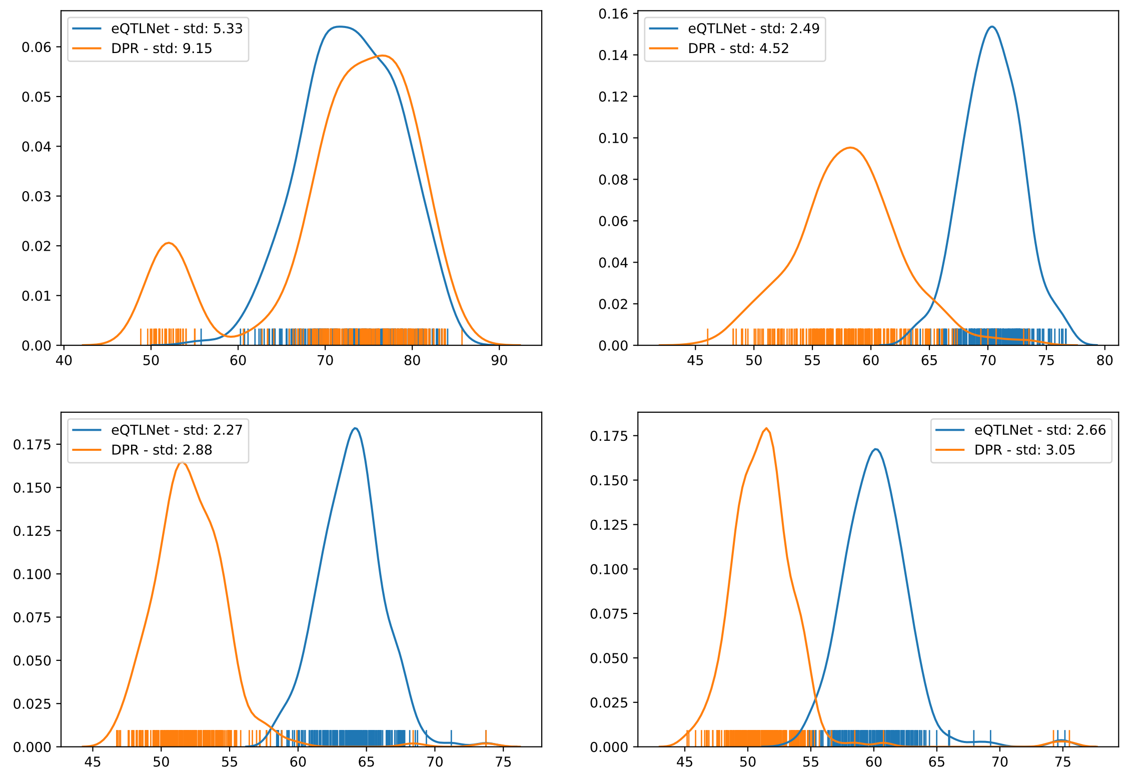


Figure 3.6: Graph accuracy distribution for different number of causal SNPs, in the absence of any residual noise.

model, used gradient clipping and a lower learning rate[5] for optimization stability. Figure 3.7 illustrates the outcomes, from which a few interesting observations can be made. First, note that when the number of causal variants increase Elastic Net performs better than the DPR. Second, despite the fact that, on average, eQTLNet surpasses DPR in all four for experiments, for the case with a small number of causal variants, the median performance of DPR is slightly higher, which is consistent with the multi-modal distribution of DPR's prediction performance (as show in Figure 3.5). In other words, for small number of causal variants, eQTLNet tends to surpass DPR by a large margin, for some samples, while for the majority of others DPR outperforms by a small margin. This suggests that for traits with low polygenicity DPR cannot be considered as a robust predictor due to notable number of

[5]The learning rates were selected according to a partial grid search.

outliers in it prediction performance, which is also observed from the distributions shown in Figure 3.6.

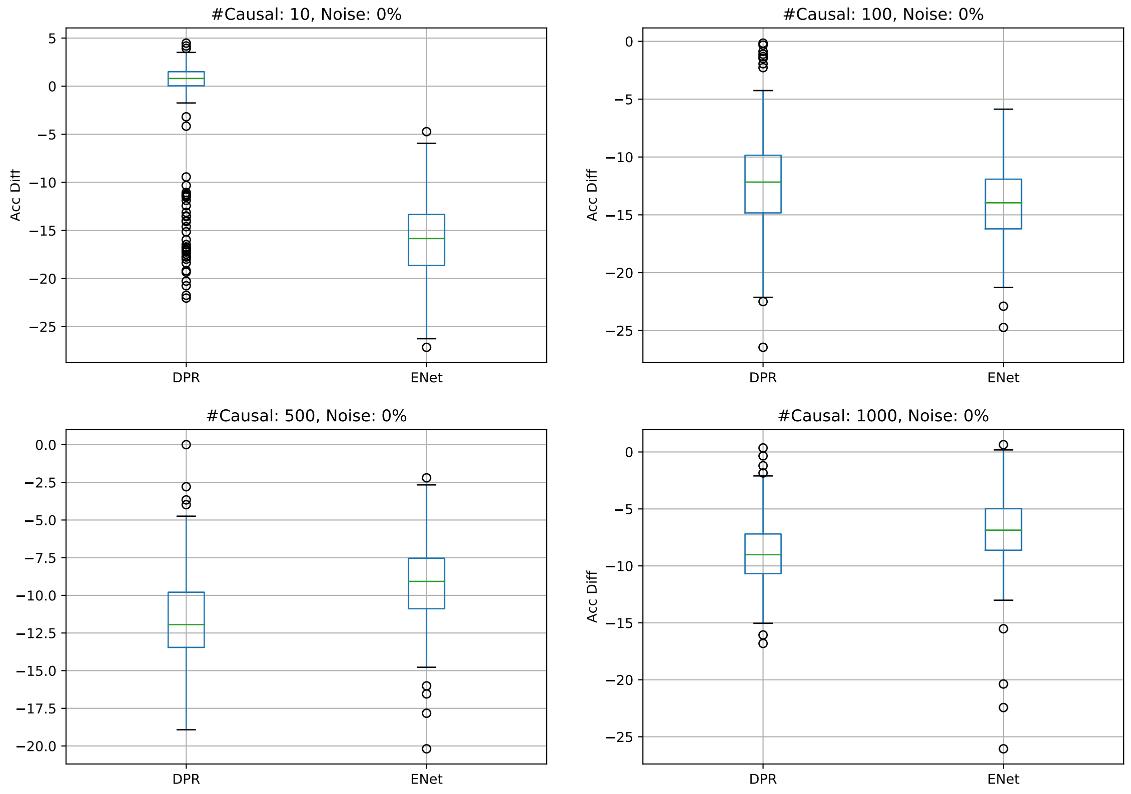


Figure 3.7: Box plots of graph accuracy improvements with respect to eQTLNet for DPR and Elastic Net. Negative values mean lower accuracy compared to eQTLNet.

So far, we have shown the superiority of eQTLNet in predicting the pairwise relation between any two samples. Often times, this is not enough and instead, we would rather to assign a rank to each individual so that we would have a measure to compare samples at the same time. In other words, we need to find a strict partial order from the noisy phenotype graphs that we have computed. To do that, we implemented the approach we proposed in section 3.4.2 to convert the noisy graphs into a consistent phenotype graphs. We used the out degree of the resulting graph as the rank of individuals, as discussed in Corollary 3.4.2.2. To make sure the reconstructed graphs are maximally consistent with the original graphs inferred from our Siamese network, we calculated graph accuracies for the new graphs and plotted the distribution of deviation in accuracy as depicted in Figure 3.8. The distribution for each dataset appears to be close to normal with mean that is only slightly (i.e. 0.1%) less than 0. This proves that our eQTLNet-ga can find a valid strict partial order of samples with a phenotype graph accuracy that is almost equal to that of the noisy graph.

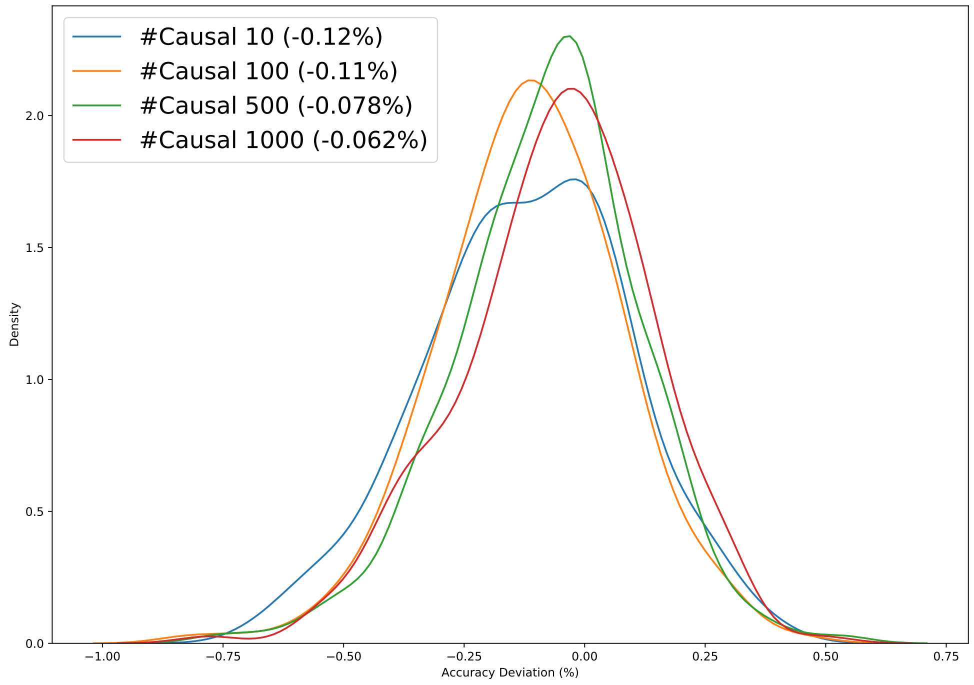


Figure 3.8: Accuracy deviation for inferred vs corrected phenotype graphs for the four polygenicity levels. The mean deviation for each experiment is show in braces and is around 0.1%.

Scenario #2: Prediction of Phenotypes in the Presence of Missing Heritability

In the previous scenario, we analyzed the performance of eQTLNet using four different polygenicity settings and under the assumption that all the phenotypic variation can be explained with the cis variants. In reality, however, this is not the case. In fact the major portion of the phenotypic variance is not explained by the cis-variants, as discussed before. To account for this missing heritability, for each individual, we added a residual term to the effect sizes of variant, drawn from a uniform distribution from the range $(-0.01x, 0.01x)$ where x is the noise level. We tried for noise levels, 10%, 20%, 30%, 50%, to assess the performance of each model in the presence of different levels of noise.

Figure 3.9 shows the performance of each the three models when the noise level and polygenicity vary. Interestingly, for high polygenicity scenarios the classical elastic net method surpasses DPR for all noise levels. For low polygenicity cases, however, DPR significantly outperforms elastic net, but the improvement margin diminishes as the missing heritability increases. In real eQTL datasets, as will be discussed, it turns out that majority of the loci fall in this region (i.e. low polygenicity and high missing heritability), leaving a small room for the baselines to compete. The second important observation from this figure is that, eQTLNet seems to outperform all other models in all the experimental settings.

We also calculated the deviation in accuracy after flipping inconsistent edges (i.e. eQTLNet-ga) to see how the addition of noise can impact the reconstruction performance. Figure 3.10 depicts distributions of the deviations for all 20 experiments. As before, the mean deviation turns out to be less than 0.1%, for a significant majority of the cases which suggests that GA is able to find relatively optimal consistent phenotype graphs via flipping the edges, hopefully in a minimalistic way. While the objective we are optimizing by randomized optimization more seeks a solution with the smallest sum of weights among flipped edges (c.f. section 3.4.2), ideally, we would want this to occur with as few as possible number of flips. Figure 3.11, shows the reversion rate, defined as the frequency of edge flipping w.r.t to the graph size, for each experiment. As suggests by the figure, the reversion is around 0.25% but increases as polygenicity and noise level increases. Finally, to see if this ratio can be improved further, we computed the improvement in the population fitness[6] as a function of generation number, when the polygenicity is set to 500. According to the figure, the improvement plateaus after about 40 generations, which suggests that no further improvement can be made if we run GA for more iterations. Furthermore, higher noise

[6]Defined as graph accuracy of the best solution in a population with w.r.t the ground truth graph.

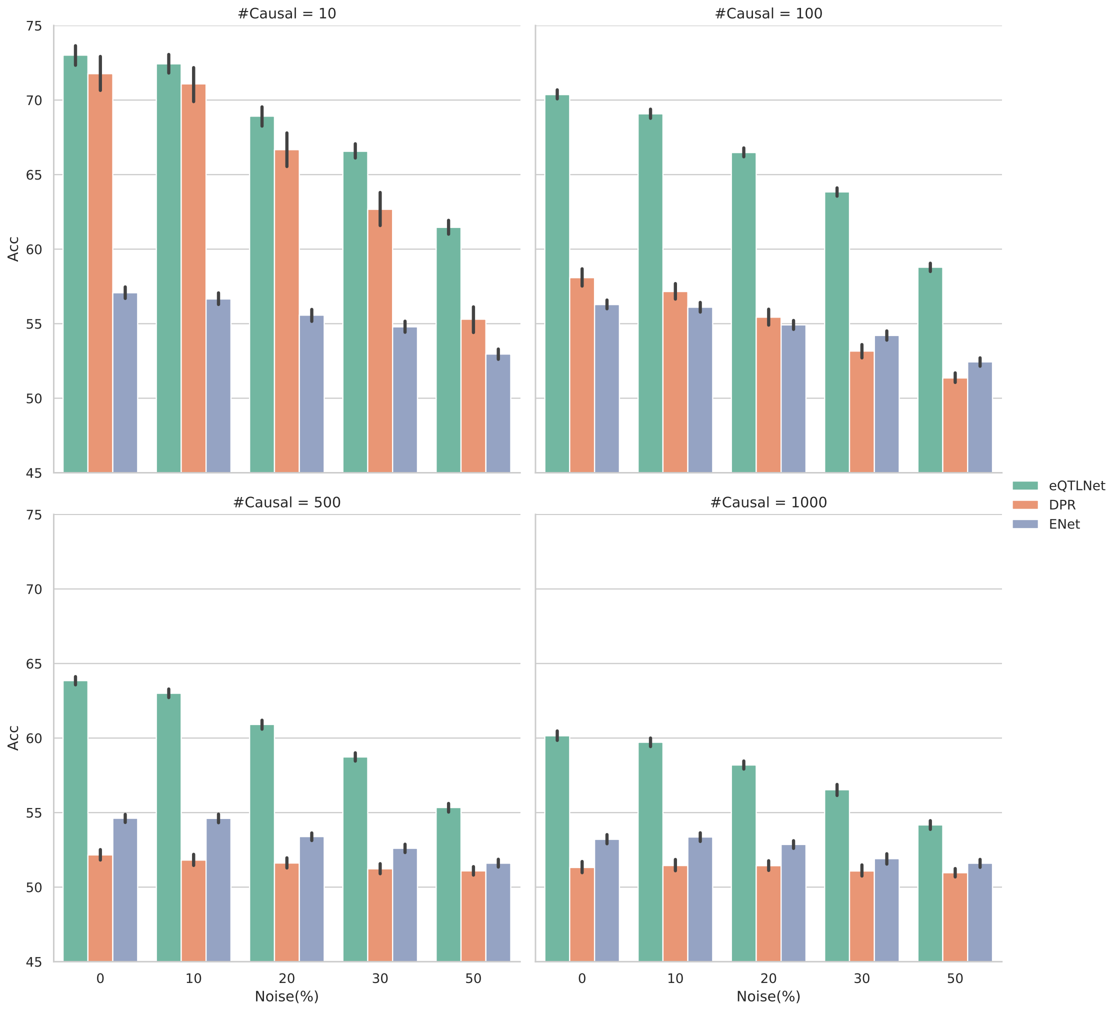


Figure 3.9: Mean accuracy of each method for 20 experiments varied by polygenicity and noise level. eQTLNet surpasses other methods on both ends of the spectrum.

levels would mean smaller observed improvements.

3.5.3 Application to Real-World Data

In section 3.5.2 we shown the superiority of eQTLNet-/ga over the two baselines on the simulated data. While simulated data allows to make a quantitative assessment of the methods, they may not represent the real data well due to their underlying simplifying assumptions. In this section, we show how each method performs on real data. Specifically, we used data from previously published papers [65, 63] with gene expression and genotype

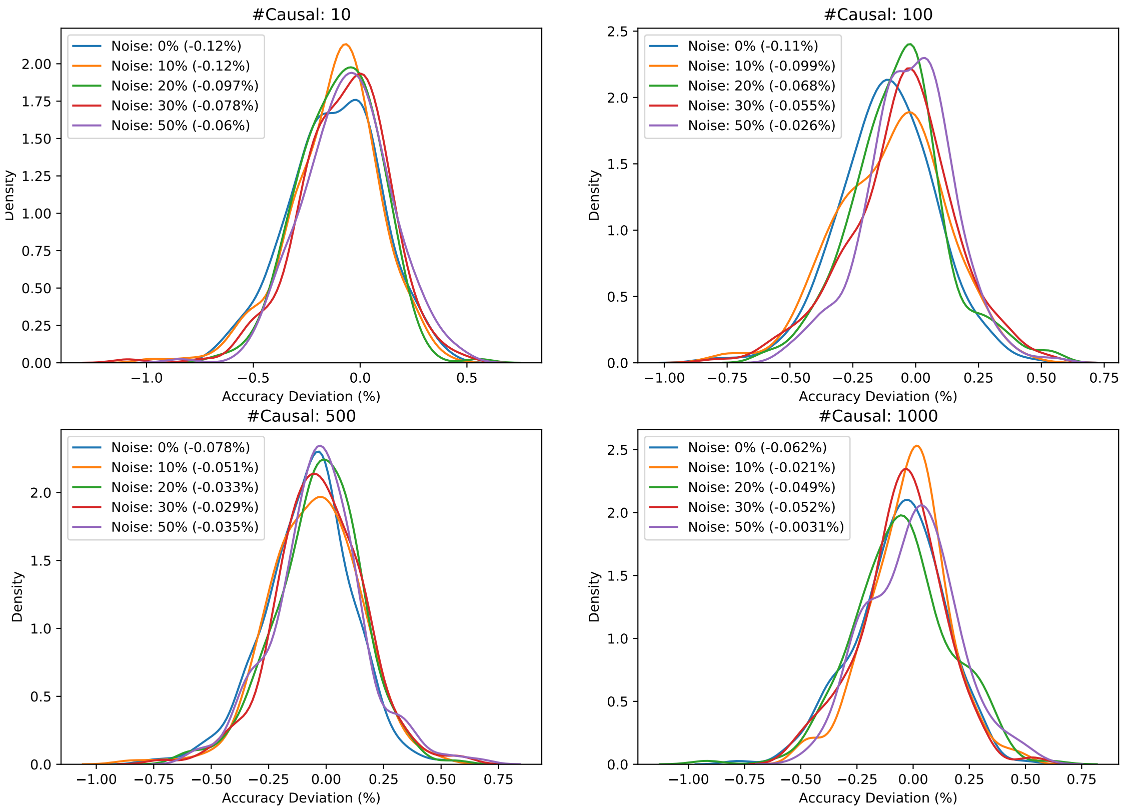


Figure 3.10: Accuracy deviation for inferred vs corrected phenotype graphs, for 20 different scenarios varied by the noise level and the number of causal variants. The mean deviation for each experiment (shown in braces) still remains around 0.1%.

data (c.f. section 3.3), randomly selected 500 loci from each and applied the three methods to them. We used batch sizes of 50, and 350 for PROTECT and CAGE_EGCUT datasets, respectively. Figure 3.13 shows scatter plots of DPR vs eQTLNet graph accuracies for the two datasets. eQTLNet and DPR performs relatively the same for loci with high prediction accuracy, while for the remaining loci, the scatter plot appear to follow a normal distribution, centered around 50% accuracy. These majority cases, could be trans-eQTL cases or those with low degree of heritability. The mean deviation in graph accuracy for PROTECT dataset is slightly less than zero, while for the CAGE dataset it is slightly more than zero. This suggests that for larger datasets eQTLNet has an advantage, though it could also depend on the complexity of phenotype-variant relation. Table 3.1 illustrates average graph accuracies over all predictions as well as the top 20% of samples in which the method per-

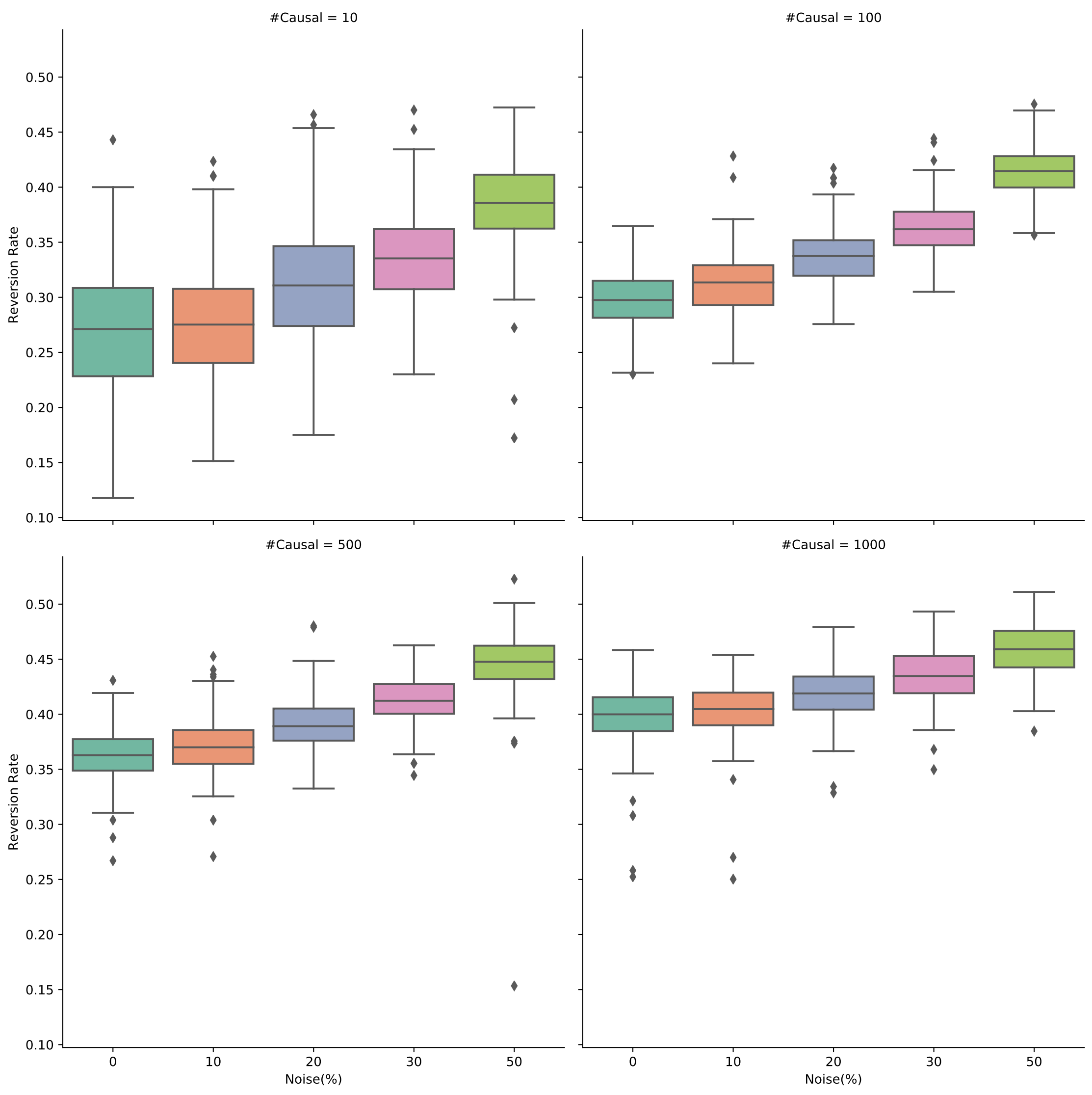


Figure 3.11: The reversion rate of edges in the noisy phenotype graphs, after optimization Genetic Algorithm. More noise levels results in a higher rate of reversion.

forms the best. Overall each method is performing better on the CAGE_EGCUT dataset, possibly due to a larger training set. For the PROTECT dataset eQTLNet is the winner model, while for the CAGE dataset, it is the runner up method, both with a small margin. Moreover, despite being an old method, elastic net is doing worse compared to DPR by only a half percent, as depicted in Figure 3.14. This indicates that there is not enough room left for improvement. As alluded earlier, this can be due to the missing heritability and

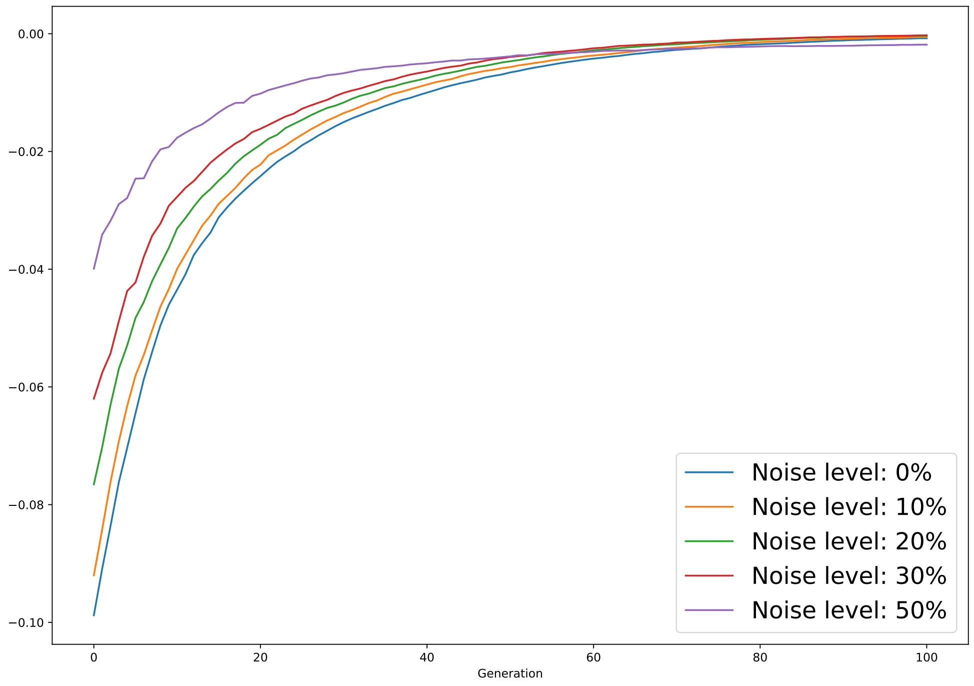


Figure 3.12: Fitness deviation from the original noisy solution.

Table 3.1: Average performance of each method on both datasets. Best results are in bold face type.

Dataset/Method	Elastic Net		DPR		eQTLNet	
	All	Top 20%	All	Top 20%	All	Top 20%
PROTECT	50.80%	57.41%	51.36%	57.99%	**51.46%**	**58.39%**
CAGE_EGCUT	52.33%	59.86%	**52.81%**	**61.47%**	52.63%	61.35%

complex relationships between variants and phenotypes. For the latter case, we might be able to devise more complex models, which require larger datasets for training.

Finally, as mentioned earlier, eQTLNet can be considered as a matrix factorization model that is trained via learning pairwise relations. Here, we shed more light on this aspect of our proposed model. Specifically, we took the representation of samples in the latent space that our model learn. To do that, we first selected two loci from the CAGE_EGCUT dataset for which eQTLNet performs well and two samples for which it is doing poorly. We then used PCA and took the top two PCA principle components for each of the four

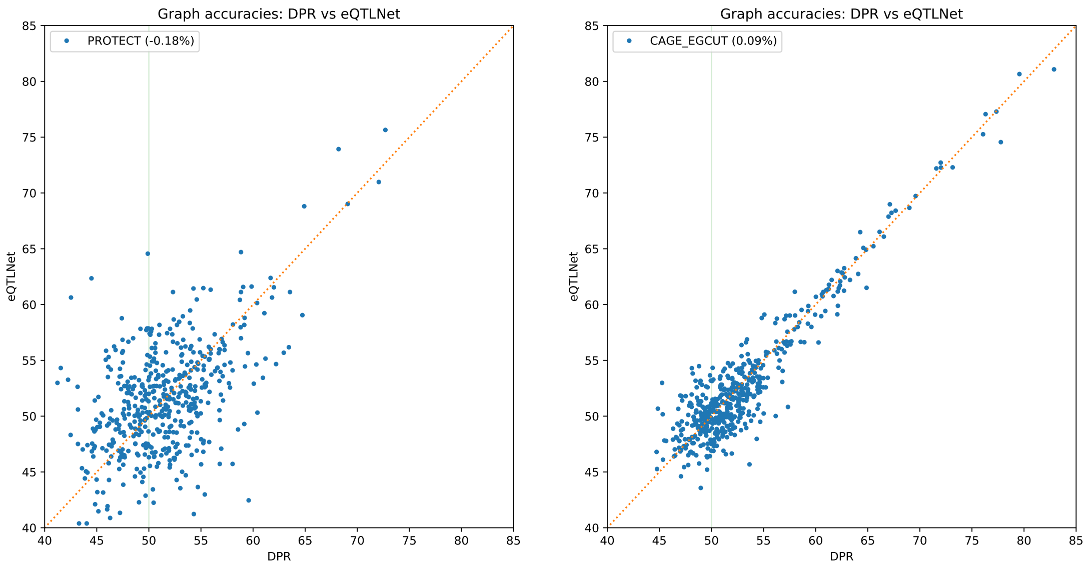


Figure 3.13: DPR vs eQTLNet scatter plots for CAGE_EGCUT (left) and PROTECT (right) datasets. Number in braces is the mean accuracy deviation.

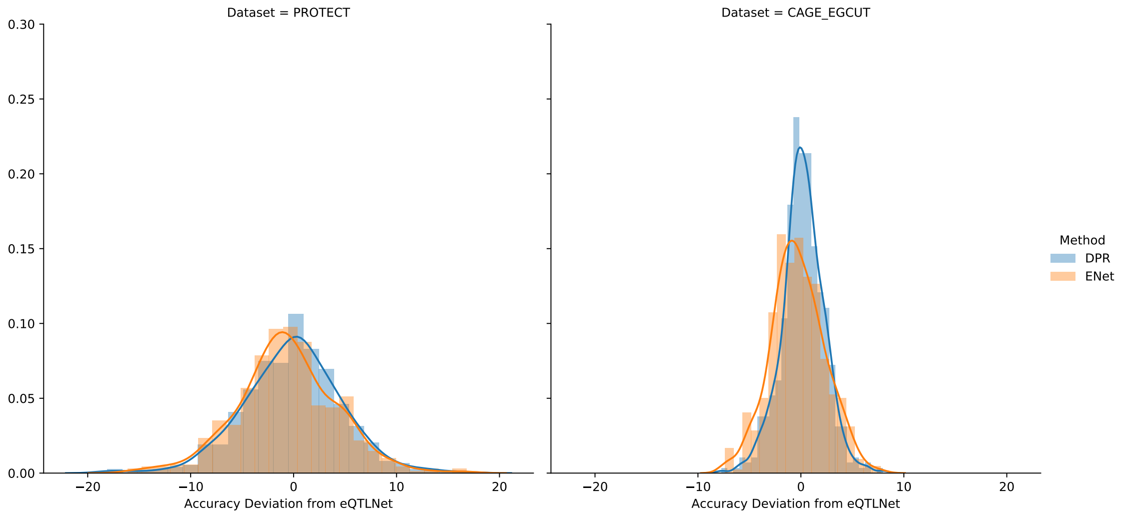
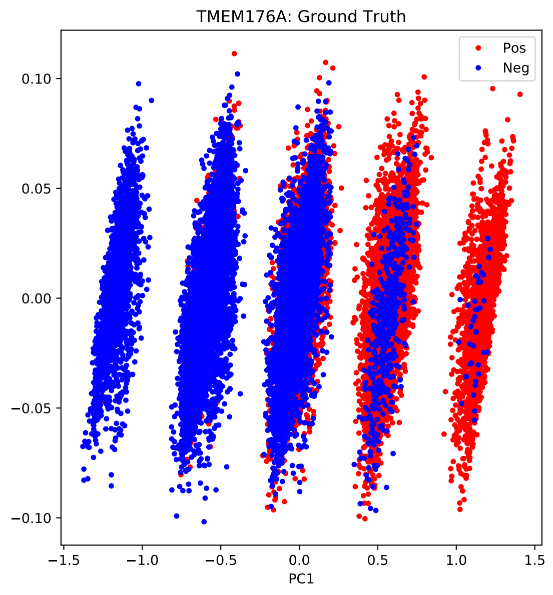


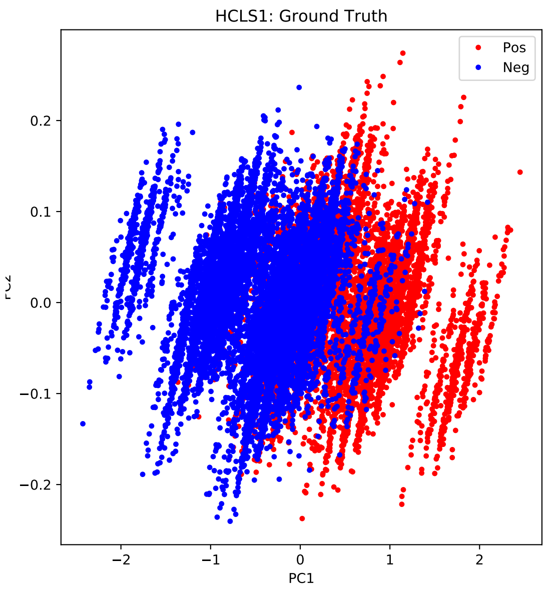
Figure 3.14: Graph accuracy deviation of DPR and elastic net w.r.t. eQTLNet (i.e. $acc_{method} - acc_{eQTLNet}$) for PROTECT (left) and CAGE(right).

samples. Figures 3.15 and 3.16 illustrate the result. Interestingly, for the two samples that our model performs well (i.e. Figure 3.15), eQTLNet grouped samples into 5 sub-types, consistent with the size of latent space. The model then assigns a positive label to two clusters and a negative label to two others. One cluster has also a mix of two classes as can be seen from figures in the right column[7]. These clusters may also reveal interesting insights about pairs of patients that belong to the same group, a fact that can be investigated in a future work. Such a structure of course may not exist for all loci. In particular, for the two samples that exhibit a poor performance (i.e. Figure 3.16) eQTLNet, no such latent space structure exists.

[7]Note that this is a representation for pairs of samples.

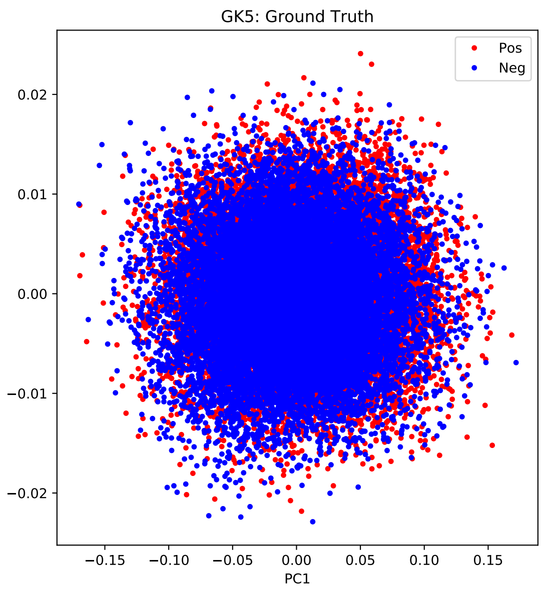


(a) TMEM176A

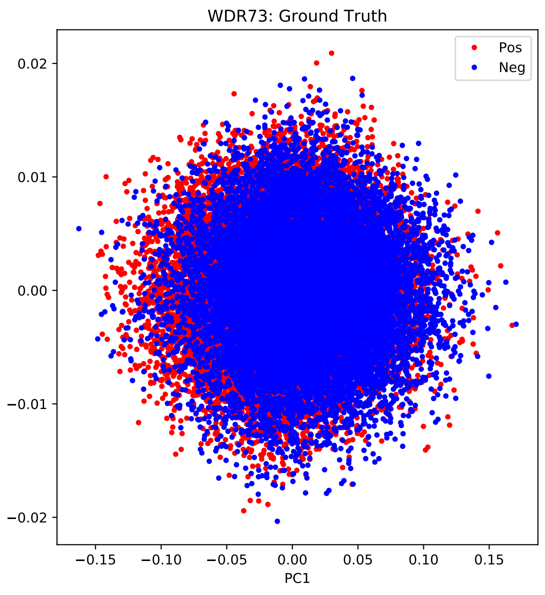


(b) HCLS1

Figure 3.15: Visualization of the two principal components of the latent space representation of two samples for which eQTLNet performs well. Colors in the left and right columns indicate the true labels, and predictions, respectively.



(a) GK5



(b) WDR73

Figure 3.16: Visualization of the two principal components of the latent space representation of two samples for which eQTLNet performs poorly. Colors in the left and right columns indicate the true labels, and predictions, respectively.

3.6 Discussion

In this chapter, we proposed a new pipeline for prediction of phenotypic ranks, composed of two major building blocks, one that inspired from the Siamese network architecture, predicted a noisy phenotype graph by learning pairwise relations, and the other which utilized randomized optimization for reducing the noisy phenotype graph into a strict partial order of samples.

To make a fair comparison, we selected the state-of-the-art DPR method [55], which has been shown promise compared to other popular techniques such as LMM [74], BVSR cite-bvsr, BayesR [75], MultiBLUP [76], and rjMCMC [77]. DPR assumes a zero-mean normal distribution on the effect sizes with its scale factor being characterized by the Dirichlet distribution. Therefore, it lifts the restriction that almost all other Bayesian approaches suffer from, namely, that the scale factor (or standard deviation) follows some predefined distribution that might not fit the real distribution well. Utilizing a frequentist model, we do not need to worry about such a restriction, as we are directly learning the effect sizes from data. Therefore, from this aspect, our model has the same advantage as DPR does. The challenge, however, is that, in general, non-Bayesian models fall short when competing with models for the aforementioned category. As was discussed in this chapter, this to some degree, can be attributed to the wrong conventional approaches in training such models. Our approach to learn relations instead of the quantity of interest, along with the Siamese-based way of training, proved that, models in the latter category can be as good as the most successful approaches in the former. We also showed that, if an individual ranking is what we desire to generate, we can use randomized optimization to create a phenotype graph from which we can compute ranks in polynomial time.

We tested our model both on the simulated and real data. We generated the simulated data under the assumption that effect sizes for each of the three genotypic states can be draw from an independent distribution and that effect sizes of the causal variants have a linear relation with the phenotype of interest. This, however, may not hold true in real datasets, which could the cause of diminished gain. Despite that, the simulated data can indeed provide a means for a more quantitative comparison of the models as the process which determines the relation between the data is readily known.

To the best of our knowledge, this is the first successful integration of a frequentist machine learning approach and randomized optimization for learning associations that performs in par or better with respective to non-parametric Bayesian approaches. In summary, the promise of our new method is due to a few key changes in the training paradigm, which are listed as follows.

1. Instead of directly learning the association between variants and the continuous phenotypes, the proposed model learns pairwise relationships between different samples. A model that learns to compare is more robust to the noise and uncertainty since small changes in the phenotypes may not readily lead to a change in the ordering of the underlying pairs.

2. Learning relations instead of a continuous value can also help correcting mistakes that may occur at the prediction stage. Specifically, let's assume that our model makes a a wrong classification when comparing a pair of samples x_i and x_j. In this case, unless the predicted score is significantly high, there is a chance that when we converting the noisy directed graph into a DAG through the heuristics explored earlier, we correct this misclassification by incurring some loss in the overall ordering fitness but in the interest of a larger gain. Such a correction does not exist when predicting for the phenotype values directly.

3. Last but not least, Siamese based models inherently benefit from larger input datasets. The fact that every pair-wise combination of input samples is counted as one valid input, makes the number of input samples significantly larger. From that perspective, Siamese models utilize an implicit lossless data augmentation, although at cost of

introducing dependency between the input samples.

3.6.1 Shortcomings and Potential Future Works

While the model proposed here offered advantages over the key benchmark method we explored, still the most noticeable shortcoming appears to be the low performance we observed through the experiments. Unfortunately, this applies to a significant majority of GWA studies. Specifically, in case of the problem of mapping eQTLs, studies have shown that the proportion of heritability that can be explained by cis eQTLs ranges from 0.20 to 0.38 [62, 78]. This raises the important question of what the unaccounted portion of the heritability, the so-called missing heritability, could be due to. A few hypotheses can be told. First, it could be that some rare variants have large effect on the phenotypes and since rare variants are not often times included in SNP array[8]. Another hypothesis, could be that, common variants of small effects are explaining for the missing heritability. And since finding such missing variants require a significant amount of data, they remain undetected, this is especially the case for trans variants. Furthermore, we almost always assume a linear relationship between the variants and the phenotypes, which does not necessarily have to be true. For example, even if we consider different effect sizes for homozygous vs heterozygous for a specific allele, that makes the relation a non-linear one which cannot be fully captured with linear models. On the other hand, training a non-linear model poses serious challenges itself and demands more data and has so far proven infeasible. Interpretable outcome is another shortcoming that is mostly attributed to the neural network based models. Often times such trained networks are considered as black boxes that may perform well with respect to the prediction of end-points of interest, but fall short in eliciting actionable and insightful rules regarding the hidden patterns. Perhaps the best we can do so far, is to resort to ad-hoc techniques such as saliency maps and other similar methods that partially address the issue.

[8]Even if they are included, still tagging rare variants through imputation to a common reference is subject to a lot of noise.

The model presented in this chapter can be potentially expanded to a non-linear and potentially deeper one, with minimal effort, by adding non-linear activation functions and layers. This might have the benefit of capturing non-linear relations between the variants, if such a relation exists. One downside is that more data will then be necessary to train such a network. With the accumulation of more and more genotyped samples, this may soon become a feasible approach in near future. Another key directions to pursue as future works are the followings.

- An important extension to the work can be through development of a hybrid model that takes advantage of the strengths of different models, in a synergistic way. In section 3.5.2 we showed that DPR performs reasonably well when polygenicity of a trait is low, where as our model significantly outperforms on the other end of the spectrum. While there is no way for us to measure polygenicity in real-world GWAS datasets, we can however, build a hybrid model by combining the phenotypic graphs derived from both models by adjusting edge weights and consequently edge directions.

- Our current model focuses only on one cohort. This is to avoid complexities of the population structure. An extension to the work could be to incorporate population structures into the model to learn more informative population-linked representations that can assist in predicting phenotypes across different cohorts. This is of course an important shortcoming of DPR and other baselines explored in the original paper. Including data from different cohorts can in fact boost the performance of phenotype prediction for individual cohorts mainly due to addition of data and thus reducing the risk of overfitting.

- Development of complex traits is often manifested through multiple tissues at different time points [79]. This suggests that the pattern of phenotypic variation in multiple tissues can disclose further insights into the the effect sizes of different variants on the phenotypes as well as the relation thereof. One potential approach to pick up such relations is through training a network adopting a multi-tasking paradigm that tries to learn different patterns from the variant data, at the same time. Multi-task learning in neural networks has recently gained attention in the Bioinformatics domain due to its success in improving the state-of-the-art performance for a number of problems [80, 81, 82, 83].

- While the running time was not a concern for us in this study, it is worth noting that training eQTLNet appears to be slower than the DPR method, which is one of the fastest programs available for GWA analysis. This slowness can be attributed to the library used for developing our model[9] as well as the need to run thousands of rounds of stochastic gradient descent optimization algorithm for training a relatively large number of weights[10].

[9]We used the Tensorflow library which is developed based on Python.

[10]For a loci with 3K SNPs the model will have to train 15K weights.

CHAPTER 4
THE LD STRUCTURE OF GENOMIC LOCI ARE ASSOCIATED WITH THE UNDERLYING VARIANTS ANNOTATIONS

4.1 Abstract

In chapter 3, we proposed a new pipeline for prediction of phenotypic ranks which was composed of two major building blocks, namely, one that was inspired from the Siamese network architecture to predict a noisy phenotype graph by learning pairwise relations, and another, which was utilizing randomized optimization for inducing strict partial orders of samples from noisy phenotype graphs. We showed that when combined together, our parametric frequentist pipeline would result in performance better or comparable to the state-of-the-art Bayesian non-parametric models.

In this chapter, we pose a new hypothesis for a different problem in GWA domain and investigate its validity computationally, using an extension of the model in the previous chapter. More specifically, here, we are more concerned about how pairs of SNPs in a given loci can linked together in light of their functional annotation, while in the previous chapter we were mostly concerned about how individuals (i.e. patients) could be compared with each other given their genotypes. In other words, in this chapter, we shift our focus from an individual level analysis, to a population level analysis, and explore the possibility of an existing connection between pairwise SNP linkage disequilibrium and their corresponding annotation. We share, for the first time, interesting insights that were not known before.

4.2 Background

The subject of investigation in this chapters is closely related to two important notions in genomics and population genetics, namely, the SNP annotation and the LD structure of genomic loci. Thus, we first provide a short explanation of each and subsequently delve into discussing our hypothesis and designing a computational model that serves as a proof of concept.

4.2.1 SNP Annotation

SNP annotation is a set of genomic features that characterize the effect or function of an individual SNP using manual curation or SNP annotation tools. SNP annotation is often acquired from available information on nucleic acid and protein sequences [84]. SNPs may exhibit different effects based on where they are positioned. For example, a SNP that is located on a coding region may confer vulnerability to a disease by shortening the expressed protein of the underlying gene (i.e. premature termination) while a non-coding variation in the promoter region of a gene can result in deviation in the transcription factor binding specificity for that gene, which can lead to over/under expression of the gene (or multiple genes) and hence disrupting some biological pathway, again leading to predisposition to a disease. There are different types of information that could be extracted from genomic and genetic context, as well as evolutionary history of the variants, some of which are as follows:

- Functional Annotation attributes functions to variants based on functional regions that include epi/-genomic signals.
- Gene-Based Annotation identifies genes surrounding or harboring the variants as such variants can potentially disrupt the protein product of genes and/or their splicing patterns.
- Knowledge-Based Annotation concerns about disruption of protein domains, or metabolic pathways, or protein-protein interactions that they are linked with.
- Transcriptional Gene Regulation considers spatial and temporal factors that can affect the gene regulation process. Mutation can affect gene regulation by altering a

multitude of elements that play a role in transcription of genes, such as chromatin state, promoter region, transcription factors, etc.

- Alternative Splicing plays a significant role in generation of different protein products from a gene. A mutation may affect the splicing process by disruption the splicing sites or the signals that direct the splicing process.
- Protein Function of a gene may be altered through variants located on the coding part of genes that change amino acid sequences of their protein products. Depending on the severity of the protein modification, the variant deleteriousness may vary. Several tools have been developed so far for analyzing the functional consequence of these types of mutations such as SIFT [85], PolyPhen-2 [86], CADD [87] to name a few.
- Other Relevant Information such as evolutionary conservation and nature selection, translation and post-translational modifications, etc. can also inform us about the phenotypic impact of variants.

4.2.2 Leveraging SNP Annotation Together with LD for Improving GWAS Power

GWA studies have been quite successful in capturing genotype-phenotype associations for a number of traits and diseases. Linkage disequilibrium, the non-random association of nearby SNPs, has been used extensively used for identification of tagging (aka proxy) variants that can explain phenotypic variances. Once, such candidate variants found, the next step is to look for the real (i.e. causal) variants by scrutinizing all variants in LD with the tagging variant, that explains disease susceptibility. Several tools and methods have been introduced to facilitate identification of functional SNPs and alleles using SNP annotations. LDlink [88] was one of the initial efforts to facilitate the aforementioned goal by giving the user a means to explore variants through interactive and visual presentation of the SNP annotations, and LD scores. Later, integrative approaches were introduced for refining predicted SNP effect sizes using SNP functional annotations. Specifically, Sveinbjornsson et al. [89] used the Variable Effect Predictor (PEV) [90] method to rate the maximal consequence of each variant on the neighboring RefSeq genes. Variants were then grouped into four categories by the severity of their impact and scores in each category were used to compute an enrichment score for that group. The enrichment scores were then used to weight the variants-trait associations. A more recent method FINDOR [91] pursues the same approach as in [89] but uses an extension of LD Score Regression [92] in light of the functional group that variants belong to. In short, for each SNP, the generate a score according to the functional groups of all the surrounding variants weighted by its LD with those variants as well as an enrichment coefficient assigned to the corresponding functional group. That way, variants in functional groups that have higher effect sizes (e.g. loss of function mutations) will impact the score of SNPs they are in LD with, more than those that belong to lower effect-size groups.

A key point to note is that, to our knowledge, all the available methods that use variants annotation for fine mapping causal variants adopt a naive way of integrating the annotation information into the prediction by using a limited set of SNP annotations (or grouping them into a small number of groups) and disregarding any potential interaction between different annotation features. In this chapter, I will use a variety of SNP annotations for investigating their potential association with the LD structure of genomic loci, in a population of several cohorts. While the utility of SNP annotations for discovering the LD structure of genomic loci is a new research question we pose, as will be discussed shortly, it is well related to the recent efforts in linking annotations to associations.

4.3 Motivation

Finucane et al. [93] have shown that the χ^2-statistic for quantifying the association of a given variant is a function of its LD with nearby variants. Later, they expanded their model to incorporate variants annotation along with the LD information corresponding to genomic regions of interest [92]. Theorem bellow serves as the basis of their proposed model as well as their recent follow-up approach [92] for computing an enrichment score that was shown to improve the power of their genome-wide association analyses.

Theorem 4.3.1. *Under a polygenic model where effect sizes are drawn independently from distributions with variance proportional to $\frac{1}{\sqrt{p(1-p)}}$ with p being the minor allele frequency (MAF), the expectation of the χ^2-statistic for any variant j can be computed according to the following equation:*

$$E[\chi^2|l_j] = Nh^2 l_j/M + Na + 1 \tag{4.1}$$

where N, M, and h^2/M are sample size, number of variants, and the average heritability explained per each SNP, respectively, and a is a term representing the contribution of confounding factors such as population stratification and $l_j = \sum_k r_{jk}^2$ is the LD score of variant j which quantifies the amount of variation tagged by SNP j.

A summary of the proof given by Bulik-Sullivan et al. [92] is provided in Appendix A. The implication of this model is that the association level of each variant is relevant to its LD with the neighboring variants. Intuitionally, this means that the association for any given SNP is a function of how well it tags other variants. Therefore, if some of those variants have potentially large effect sizes, it will be accounted for. In light of this argument, they further expanded Equation 4.1 to include the annotation of variants by clustering annotations into multiple functional categories such that LD to a category that is highly enriched for heritability contributes more compared to a category with a lower level of heritability enrichment. Hence, Equation 4.1 generalizes to,

$$\mathbb{E}[\chi_j^2] = N \sum_C \tau_C l(j, C) + Na + 1 \tag{4.2}$$

with τ_C being the enrichment score of functional category C.

There are a few subtle observations that can be made from Equation 4.2, as follows:

- Perhaps the key message, that is most relevant to this chapter, is that for any variant, j, the expectation of χ^2-statistic does not depend on the phenotype values, and hence, the model states that the phenotype-agnostic portion of heritability contribution of variants is a direct function of the pairwise LD and the underlying enrichment cluster[1]. In other words, irrespective of the trait or disease of interest, we can place some universal prior on the SNP variants based on the foregoing elements. This suggests that, there exists useful information that we can elicit from SNP annotations, and the haplotype structure of loci, without a need to include phenotype values.

- The categorization of functional annotation in FINDOR is performed naively. A variant might belong to several categories. A better approach would be to consider a some representation in some latent space for each annotation. In other words, if we map each annotation to a point in this space in a way that annotations that are behave similarly are close to each other in that space, there won't be a need for such ad-hoc clustering. Ideally, we should be able to learn such a representation using the same data that is available for training, i.e. the LD matrix and variants annotations. This poses the question of whether or not the LD matrix is related somehow to this representation, a pivotal question that is the basis of DeepLD, the model proposed in this chapter.

- Finally, Equations 4.1, 4.2 are derived under the additivity assumption of variants effects. We can seek for a non-linear mapping if we relax this assumption.

4.4 Dataset and Data Preparation

We retrieved genotypes data from the CAGE study dataset [65, 66] and used 13 genes according to a recent study [94] for which evidence of strong regulatory effects were found and extracted all the variants that are located on them or within their 100K bp flanking re-

[1]Disregarding the population stratification for now.

Table 4.1: List of cohorts from the CAGE study used in this chapter

Cohort	Short Description	#Individuals
CAD	Unknown acronym	147
CHDWB	Centre for Health Discovery and Well-Being	439
EGCUT	Estonian Genome Center, University of Tartu	1065
MOR	Morocco	188

gion. Table 4.1 shows the four cohorts from the CAGE dataset that we used for our analysis along with the size of each, and Table 4.2 lists the selected genes along with the length locus selected for each. We randomly selected 8 loci and used them for training, leaving the remaining 5 to evaluate our proposed model. For SNP annotations, we used the data from the study Combined Annotation Dependent Depletion (CADD) [87], which contains upto 63 genomic features for each variant. CADD is a tool for rating deleteriousness of SNPs and indels in human genome. Using different genomic features, CADD classifies neutral SNPs from potentially deleterious ones. The set of neutral variants that it was trained on, contains those variants that have been fixed or nearly fixed during the human history of evolution from the ancestral human-chimpanzee genome. These variants, if not neutral are at most weakly detrimental as they have survived during millions of years. For the negative set, they generated simulated de-novo variants that can be potentially deleterious as they have not been successfully transfered into human genome, and in some cases, neutral.

Because CADD integrates different sources of annotation for SNPs, using its dataset will allow us to learn a holistic representation of SNPs via taking all available pieces of annotation into consideration. We removed all naming features, categorical features with large cardinality, and features with almost all entries missing. We normalized all continuous features and converted all remaining categorical features into one-hot coded features, generating altogether 205 features, as listed in Appendix B. For LD scores, we used plink 1.9 software [95] and used –r2 square option to generate the r^2 LD score from the genotype data.

Table 4.2: Gene Lengths (including the flanking regions) in base pair

Gene	Length (bp)	Train
FOLR3	418,514	✓
CPVL	598,721	✓
NFXL1	466,389	✓
SDCCAG3	404,730	✓
FCN1	407,938	✓
MFSD6	500,256	✓
LILRA3	401,962	✓
TMEM50B	444,542	✓
PLCL1	1,166,558	
AMFR	463,740	
CISD1	418,970	
S100A12	400,678	
PAM	676,201	

4.5 DeepLD: Variant Mapping Using SNP Annotations and LD Statistic

In Section 4.3 we discussed the motivation for investigating a potential relationship between genomic loci LD structure and SNP annotations of the underlying variants. As alluded there, this relation would be non-linear had we relaxed the additivity assumption which is a default assumption in almost all GWA studies. We also noted that we might be better off learning a representation for each variant that integrates all available annotation features and thereby obviating the need for a naive binning of functional groups. In this section, we present a non-linear model for learning a representation of SNPs by their annotation. While, this representation can be subsequently integrated into a GWA model to increase prediction power, our focus here will be on the representation itself, and whether it is possible to find one using solely the annotations and the LDs between SNPs. In other words, we won't use trait values for that purpose, and as a result, if such a relation turns out to exist, it will be applicable to almost all traits.

Like the previous chapter, the cornerstone of our model is a by-product of the Siamese architecture, as discussed before. However, with two key differences. First, our network will be comprised of three parallel components with tied weights, for comparing pairs of SNPs together. In other words, we are concerned about whether a pair of variants (rs_i, rs_j) is more favored to (rs_k, rs_l) given variants annotations. Our hypothesis is that variants annotations can inform about the LD structure of the loci harboring them. Second, this relation, if it exists, can be a non-linear one, hence, we wont restrict our model to the linear hypothesis space.

Figure 4.1 depicts the architecture of our proposed model, DeepLD. It is comprised of two separate Siamese network, stacked on top of each other. The objective of the first network is to generate the representation that we detailed in Section 4.3, while the purpose for the second network to learn a distance between representations that is in harmony with the LD of score of a pair of SNPs. Therefore, given a triplet of input variants, $< rs_1, rs_2, rs_3 >$, the combined network is expected to learn predict if the first pair, (rs_1, rs_2), is in higher LD compared to the second pair, (rs_2, rs_3).

The first Siamese network, takes the preprocessed annotations for three variants and generates the first level representations. This component is composed of two Fully Connected (FC) layers with 20, and 10 neurons, respectively. We used Rectified Linear Units (ReLU) activation functions for the first FC layer, while for the latter we did not use any non-linearity. We also used batch normalization modules [15] before we apply activation functions[2].

Once the representations are produced, we compute element-wise squared deviation between the outputs of the first and second modules, as well as the outputs of the second and the third ones. This serves as a distance metric in the latent space of variants representations, so that when two representations are close to each other, it leads to a high score as the output of the second network. A key point to note is that, one strong feature that affects the LD level between a pair of variants, is their distance in base pair from each other. The higher the distance, the lower the likelihood of being part of the same haplotype. As we

[2]We got a higher performance when placing the batch normalization module before the activation, despite being recommended otherwise, in literature.

train our network on the basis of comparisons between LD values, it would make sense to disentangle the impact of the distance from the representation learned for each annotation. To do that, we will also integrate the logarithm of distances as an input to the second stage Siamese networks.

Finally, the second Siamese network generates the output features for each pair of SNPs and subsequently computes the logit value for the overall classification task. We trained the network, end-to-end, using gradient descent optimization and the cross-entropy loss. To partially overcome any potential overfitting, we also added an L_2-norm regularizer to the model. Furthermore, in the training phase, we used batches of size of 3000 and enforced a filtering criteria according to which we excluded triplets whose pairwise LDs were within half standard deviation of all LDs in the dataset (i.e. $|LD(rs_1, rs_2) - LD(rs_2, rs_3) < \frac{\sigma}{2}|$).

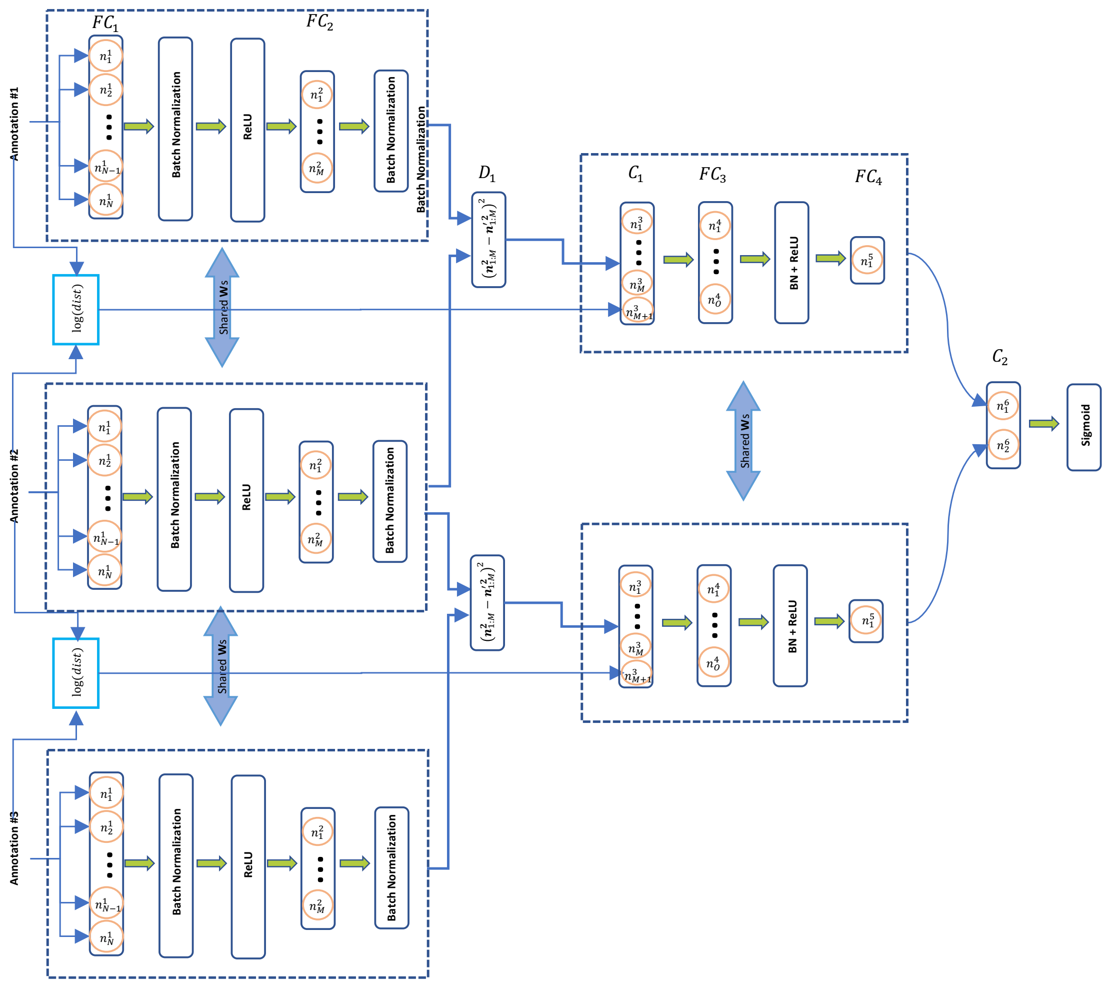


Figure 4.1: The architecture of DeepLD: The weights are shared between daughter networks in each of the two major components. Positional distance is added to the representational features learned by the first two layers.

4.6 Analysis of Performance, Convergence, and Robustness with Respect to the Size of Training Sets as Well as Triplet Distances

An important questions that is often asked by the machine learning theory on trained classifiers is about the Bias/Variance trade-off [96], which is also related to the capacity [97] (i.e. comprehensiveness) of the hypothesis space that the designed model is defined in. In the context of our SNP modeling application, we are concerned about a balance between the amount of data that we have, the complexity of our model and whether or not a meaningful pattern can be learned from data.

We used random batches of size 3000 pairs from the 8 loci randomly chosen for training (see Table 4.2) and trained the network for multiple iterations until model convergence takes place. Figure 4.2 depicts the loss and accuracy plots during the course of training, and Figure 4.3 shows the validation accuracy at the same steps of training for each test loci, individually. A few interesting observations can be made from these figures. First, it appears that for our training task the training accuracy is in complete harmony with validation results. This suggests that the amount of data we are using for training is sufficient to achieve a generalizable model. Second, learning pattern appears to be consistent across different loci, this suggests that SNP annotations have a consistent and universal impact on the the LD structure of genomic loci. In other words, a portion of the LD structure of genomic loci, can be attributed to the annotations of variants. This portion while it varies from loci to loci, remains significant across the genome. Lastly, observe that evaluation of trained model on exhaustive set of variant triplets is a highly infeasible task. On the other hand, as is suggested by these figures, a couple of hundred training steps is enough to get a stable model. That raises the question of how many samples will be enough to get reproducible evaluation results. In light of that, we trained DeepLD on the training set until convergence, and then took individual loci and evaluated the trained model by randomly drawing triplets from the space of all possible inputs. Figure 4.4 shows the performances we get when evaluating on sets of size 100 to 1000,000. From the figure, it turns out that, for all cases, 10,000 samples is sufficient for generating a reproducible estimation of the performance, which has important implications when scalability is a concern[3].

As the next experiment, we asked how much of the predictive power can be attributed to the variants annotations rather than the distance feature which is fused into the second Siamese net. This is important as the major objective of DeepLD is to learn a meaningful representation of variants rather than predicting the LD. We evaluated the performance of

[3]We often require genome-wide application of such models.

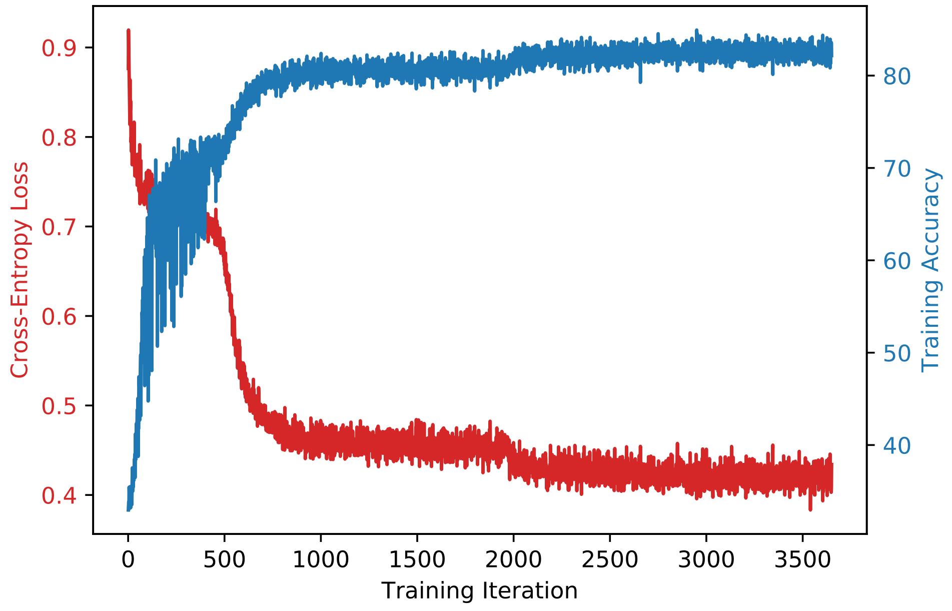


Figure 4.2: Training cross-entropy loss and accuracy.

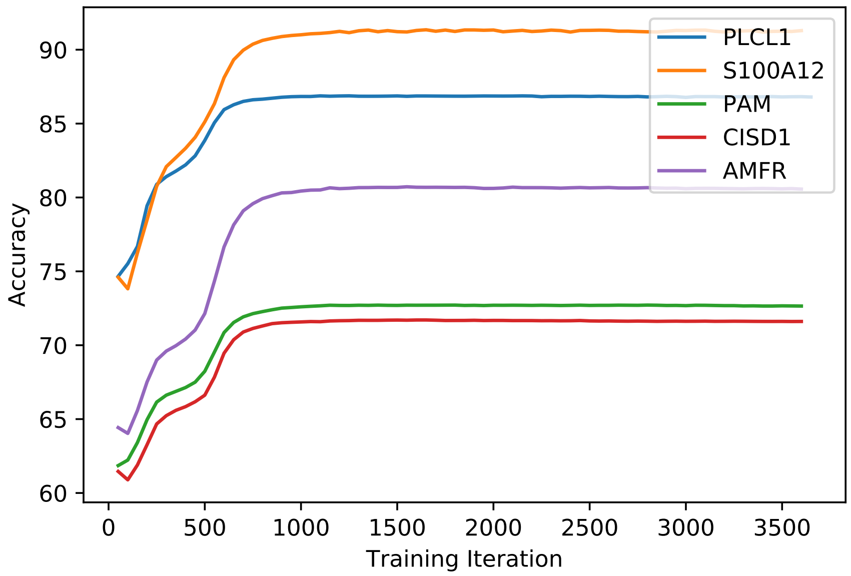


Figure 4.3: Validation accuracy for genes: PLCL1, S100A12, PAM, CISD1, AMFR.

the trained model by restricting the set of variant triplets to those that fall within windows of limited lengths. Figure 4.5 shows the prediction accuracy, for different window sizes. We evaluated windows of size as small as 1000 base pair, upto 200,000 base pairs. As expected, the performance steadily increases as we increase the window length. This is because a pair of SNPs that are closer than another pair, are more likely to be in LD with each other, than

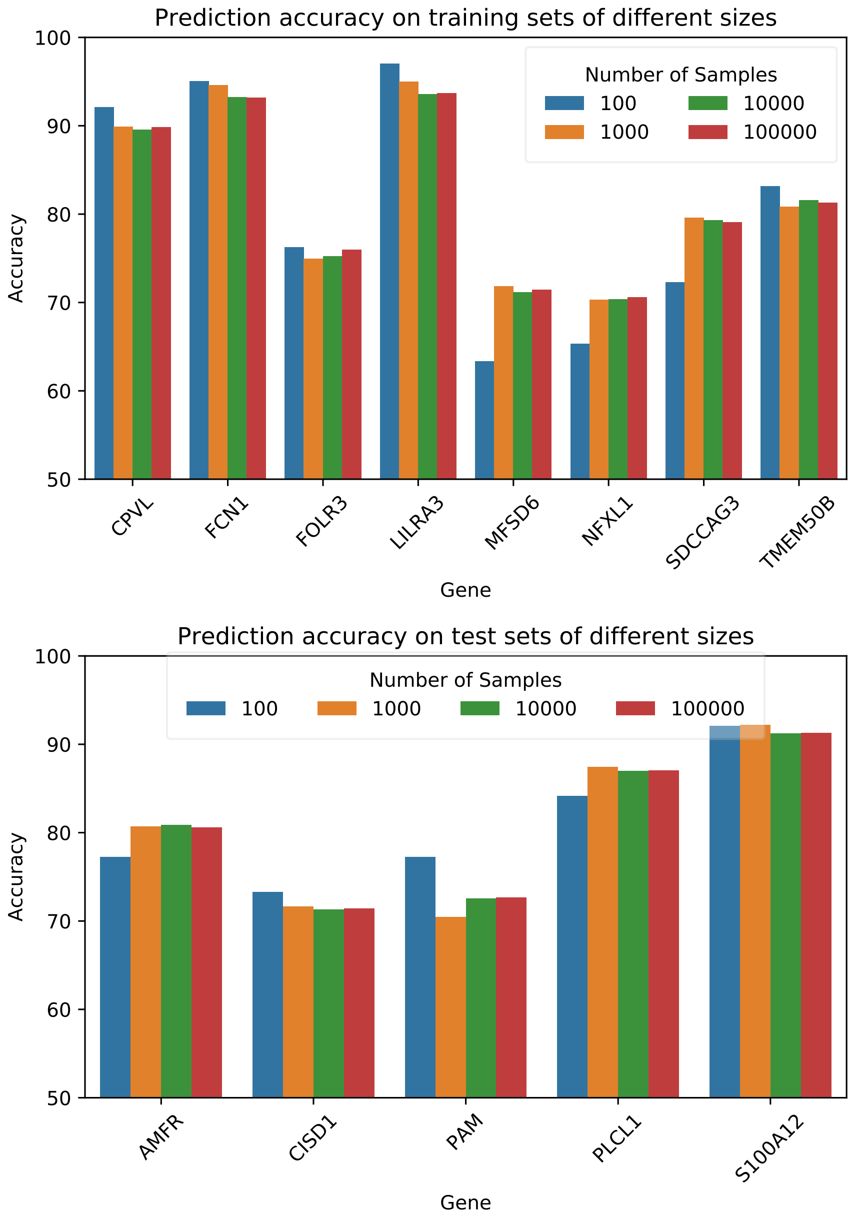


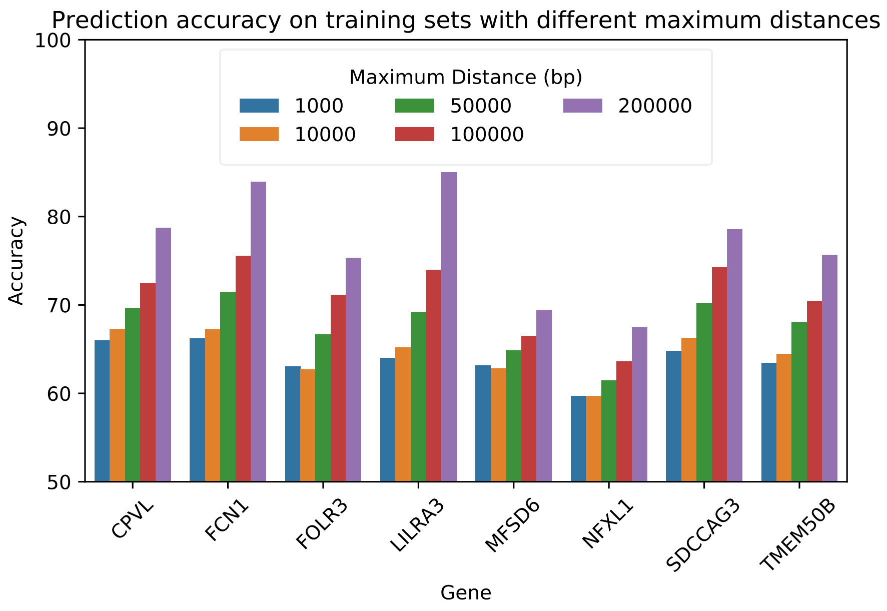
Figure 4.4: Performance variation as a function of the size of sampled triplets.

Table 4.3: DeepLD prediction accuracy on the biased and unbiased (i.e. no margin enforcement in LD absolute deviation) sets.

Gene	Accuracy (low-high LD triplets)	Accuracy (all triplets)
PLCL1	86.87%	66.21%
AMFR	80.68%	61.75%
CISD1	71.70%	61.22%
S100A12	91.22%	67.47%
PAM	72.69%	59.78%

the latter. However, when the window size is limited, this leverage no longer exists. In the extreme case where SNP pairs are located in a very short region of 1000 base pairs, the log distance between the variants are almost the same, and the classification relies solely on the annotation representations. The interesting observation that can be made from this figure is that, even when the window size is restricted to 1000 base pairs, the performances on the test loci appear significant. This is especially a remarkable performance if we appreciate the fact that genomic annotations we are using do not encode any information with an established relation to the LD structure of genomic loci.

In our analysis so far, we focused on comparing variant pairs with corresponding LDs being different from each other by a $\frac{\sigma}{2}$-margin. This was to reduce the impact of confounding factors such as not incorporating features that are driving forces in shaping genomic LD structure. As a final analysis of robustness, we asked whether DeepLD can still be effective in a more general setting, that is, when we evaluate it on an unbiased set of triplets. Note that, we are still training on the restricted set, to avoid learning spurious relations. Table 4.3 lists model performance when applied to the test loci, for both biased and unbiased sets. As can be seen from the table, while there is a notable drop in accuracy for the unbiased set, it still remains significant, which is consistent with the observations from the previous experiment.



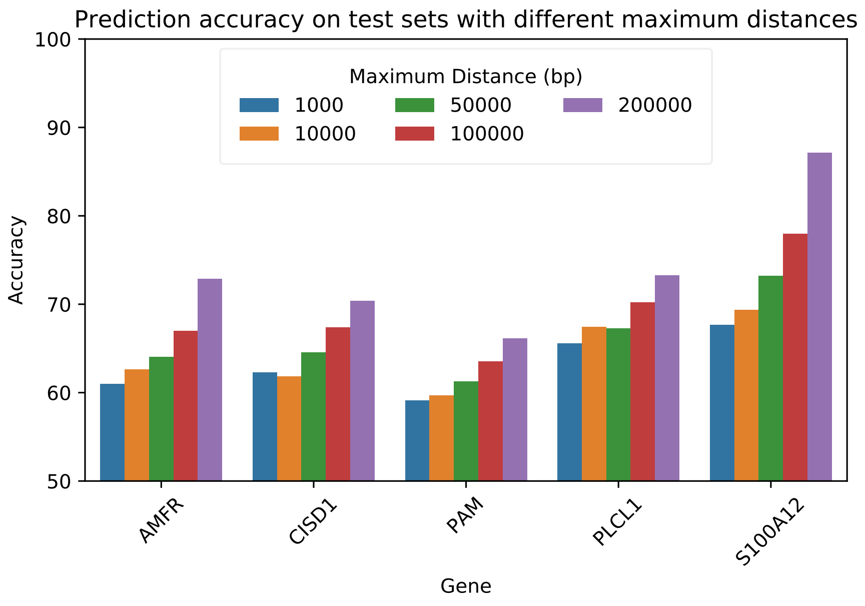


Figure 4.5: Performance variation as a function of the maximum distance between triplets.

4.7 Using DeepLD for Predicting Similarity Between Pairs of SNPs

As discussed in previous sections, DeepLD is able to compare triplets of variants through learning features that are informative for selecting high-LD pairs from the low-LD ones. While this by itself is an interesting predictive task, often times, a more important question to answer is whether or not we can predict the similarity between pairs of SNPs, instead of comparing SNP triplets. In this section I will address this key question by loading the trained network, selecting one of the two neurons generating logit values in layer FC_4, and removing any other parts not contributing to the neuron's output. Notice that the distance layer D_1 in DeepLD (see Figure 4.1) and the LD-guided learning process shapes the output of this neuron to serve as a annotation informed similarity metric.

We generated all possible pairs of SNPs and annotations corresponding to each, and fed the resulting dataset into the sub-network above.We subsequently trimed and normalized the scores and visualized the normalized predictions using heat maps. Figures 4.6-4.10 shows the visualization of both the LD structure and the normalized similarity for the test Loci, using the popular LDheatmap library for graphical display of pairwise linkage disequilibria between SNPs, developed with R [98]. Interestingly, upon comparing the similarity scores to the LD structure of the test loci, we can immediately see consistent patterns. In case of the PAM locus, the learned metric is able to pick up even subtle patterns. For this locus, all three major and one small haplotype blocks and the corresponding boundaries have been correctly recognized by our model. One other minor haplotype block has also been merged with the bigger block next to it. Such an accurate prediction is remarkable considering the fact that we are learning from features with no established relation to the underlying loci LD structure. For locus PLC1, we can make quite similar observations, even though the boundary between the two haplotype blocks shows an overlap. The can also make a quite similar observations for the other three loci, although the predicted patterns appear to be noisy.

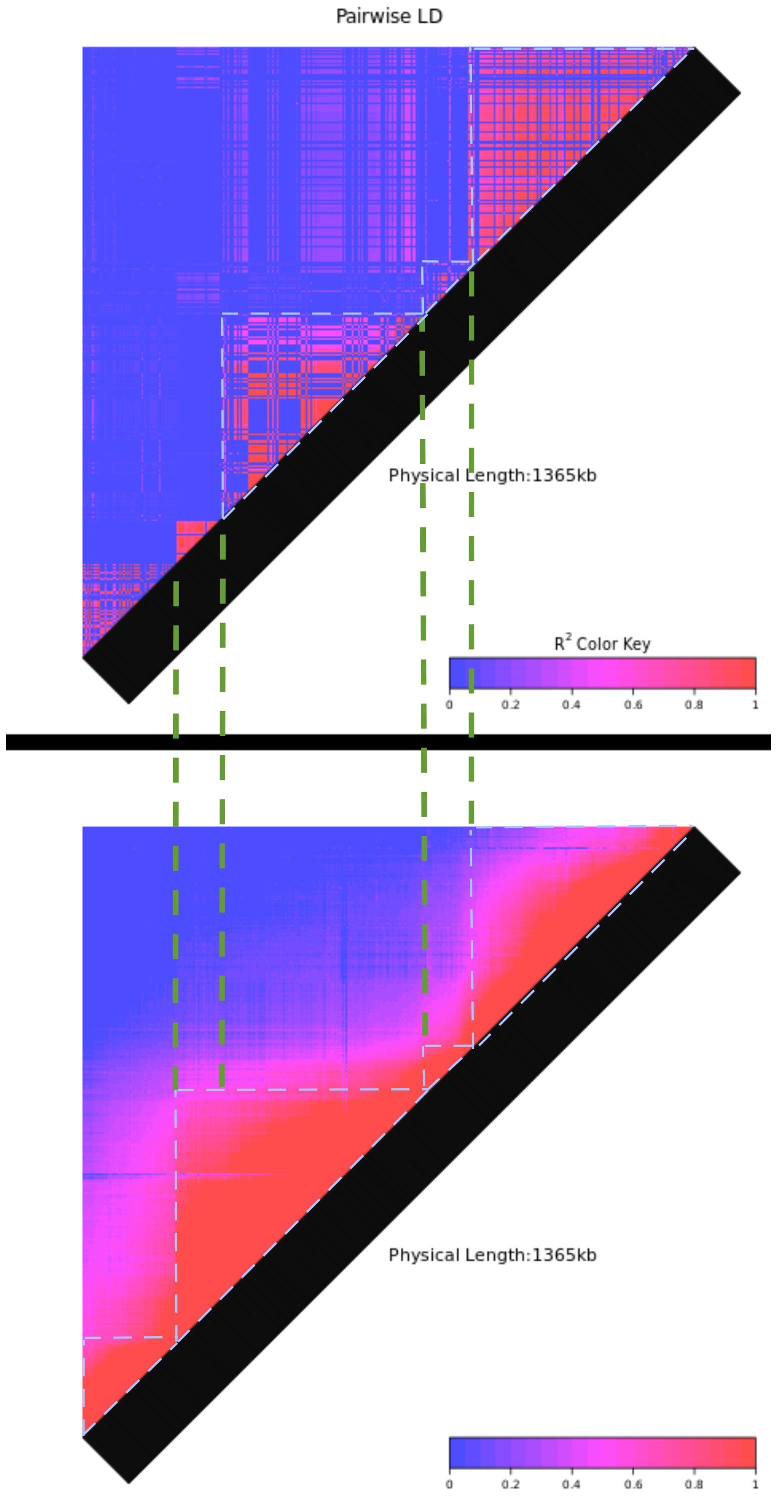


Figure 4.6: Heatmap plot of the learned similarity metric for gene PAM.

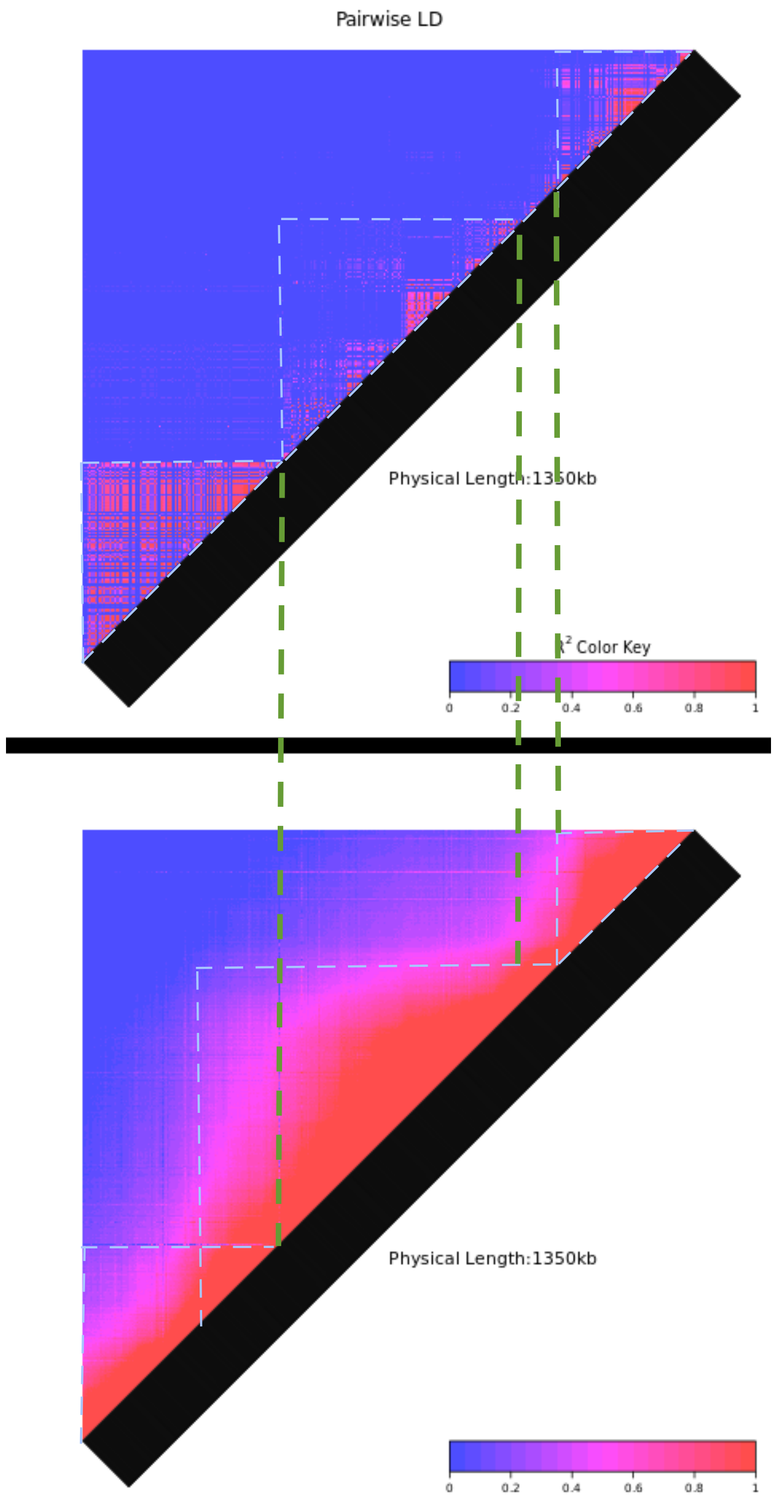


Figure 4.7: Heatmap plot of the learned similarity metric for gene PLCL1.

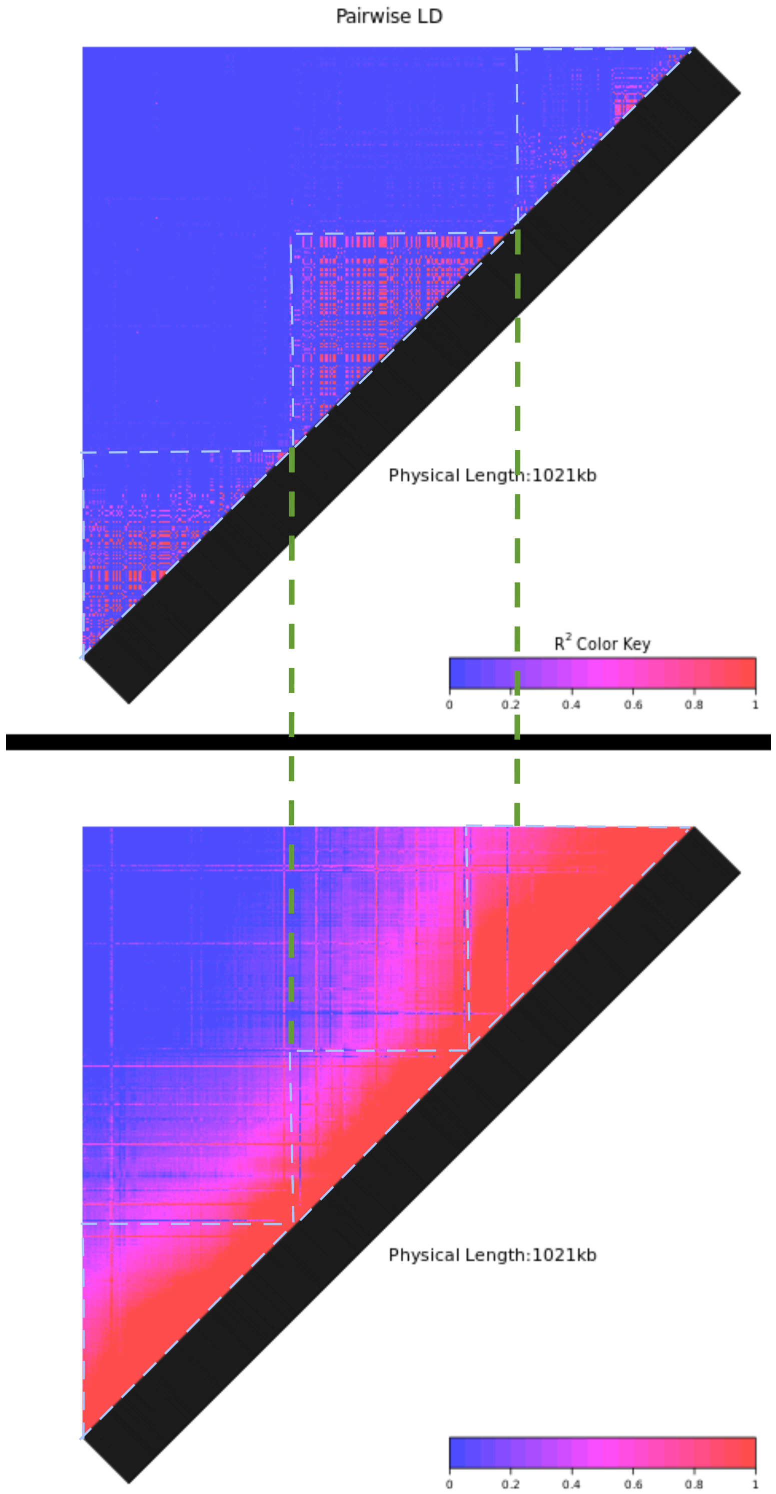


Figure 4.8: Heatmap plot of the learned similarity metric for gene AMFR.

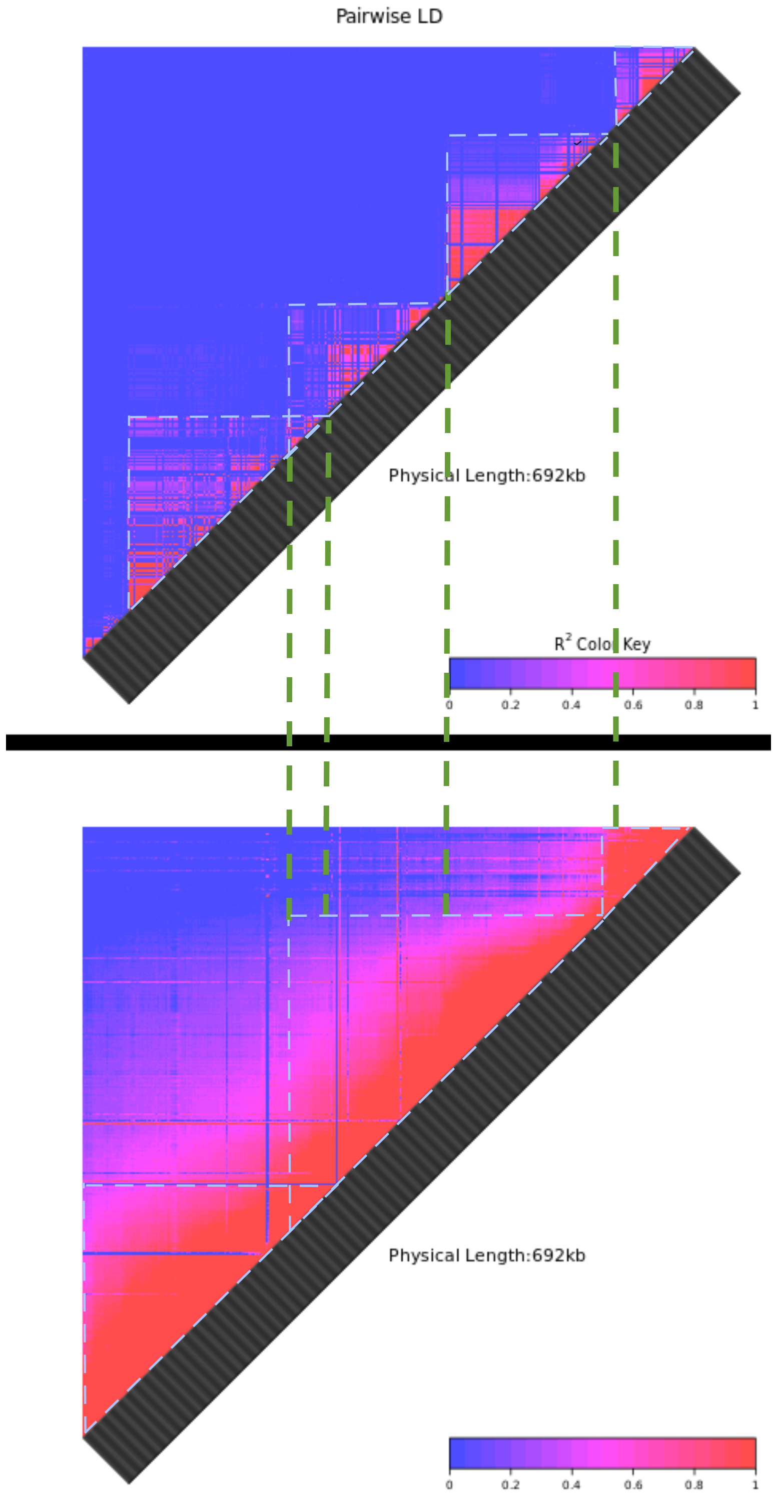


Figure 4.9: Heatmap plot of the learned similarity metric for gene S100A12.

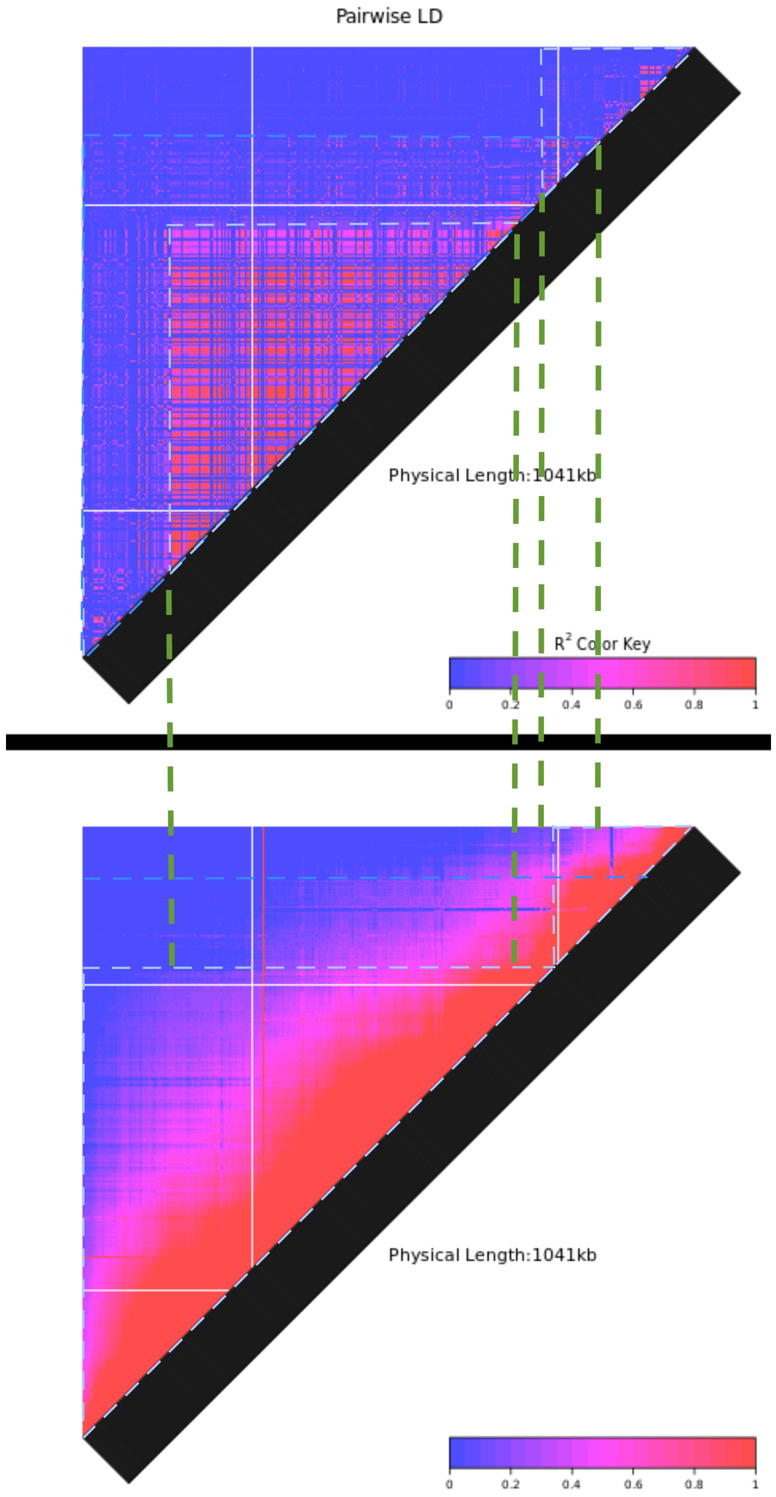


Figure 4.10: Heatmap plot of the learned similarity metric for gene CISD1.

4.8 Evaluating SNPs Similarity Metric: A Quantitative Comparison

In the previous section we visualized the similarity metric learned by our model and showed an interesting correlation between the predicted patterns and the ground truth. However, the analysis there was quite subjective. In this section we compute per-SNP LD which is defined as the sum of pairwise LDs between any given SNP and the SNPs around it. Likewise, we compute the per-SNP similarity from the prediction scores, for each locus. The goal is to investigate any positive relationship between the two and to see whether or not such a relation is statistically significant.

Figure 4.11 depicts the scatter plots of the per-SNP LD and similarity scores, along with the coefficient of determination for the regression line and the Pearson correlation coefficient. The p-value for the Pearson correlation is noted in braces. AMFR, CISD1, and S100A12 show a positive Pearson coefficient while for the PAM and PLCL1, the coefficient is negative. In both cases, the problem can be attributed to merging of haplotype blocks, by DeepLD.

For PAM, this appears to be due to merging a small haplotype block with the big neighboring one which leads to amplification of the scores for SNPs that are located in the boundary areas. Likewise, for the PLCL1 gene, three middle haplotype blocks are joined into one. For this case however, there appears to be some LD between SNPs in different blocks, which supports DeepLD's decision for merging. Overall, projecting the two dimensional pairwise LD/similarity into one dimension wipes out useful information and may not be the right way of evaluation.

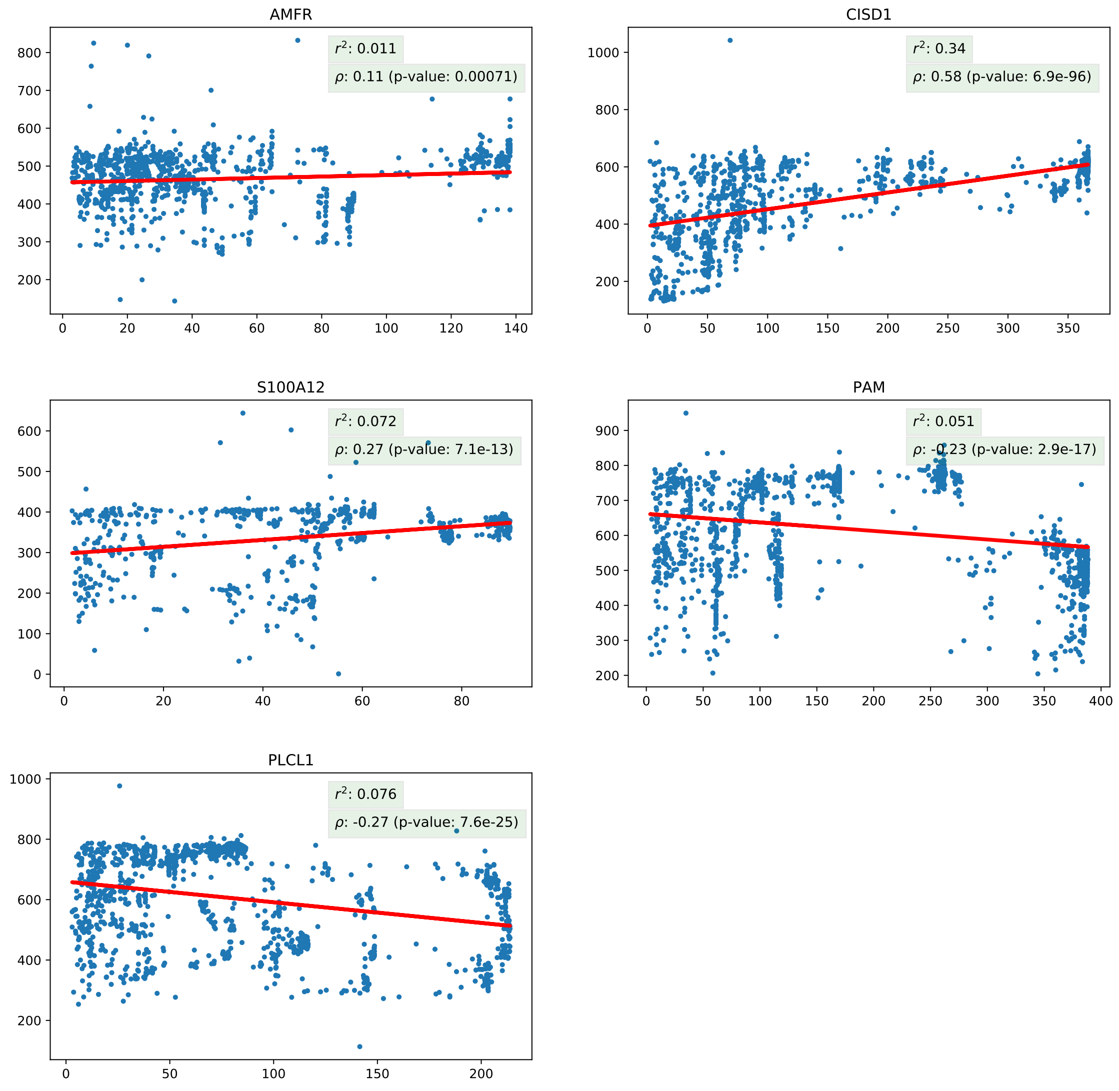


Figure 4.11: Scatter plots of LD scores vs sum of normalized similarities and the regression line underlying them. Legend shows the Pearson correlation coefficient and coefficient of determination for the regression score. X and Y axes show the per-SNP LD, and the per-SNP similarity scores, respectively.

4.9 SNPs Representation by DeepLD

As discussed in Section 4.5, the role of the first Siamese network in DeepLD, was to learn an informative representation for each SNP, given its annotation. To test this hypothesis, we took the trained model, and removed all its modules but the first daughter network in the first Siamese network and used the neurons in layer FC_2 as the representation of the SNPs. Note that at this level, the distance feature is not fused into the model yet (c.f. Figure 4.1). Next, we fed the annotations for SNPs of a locus, in this case PAM, into the model and stored the predicted features from layer FC_2 for later visualization. Subsequently, we used t-SNE [99] to map the stored features into two dimensions and distributed samples into 14 bins according to their position on the locus. We colored samples in each bin with a different color.

Figure 4.12 depicts the resulting coloring of each SNP in the PAM gene. The expectation is that if two SNPs are inside the same haplotype block and they are close to each other on the genome, we should map them to the same label or a labels close to each other as they should localize in the latent space learned by first sub-network of DeepLD, and indeed, this is what we can see from this figure. Note that bins boundaries are not opted based on the haplotype boundaries, but rather, based solely on SNPs positional order. Hence, artifacts do exist in our coloring scheme. In spite of that, we can clearly see a localizing pattern in this figure, consistent with observations we made in the heat map experiments. To see a zoomed in view of the figure, Figures 4.13a and 4.13b show only samples that belong to bins 1,5 and 2,7, respectively, for which a clear separation pattern can be seen. These observations suggest that we can use any clustering algorithm to group SNPs such that SNPs that fall in the same cluster can be assigned similar normalization coefficients as opposed to the ad-hoc approaches adopted in GWAS studies such as [91, 89].

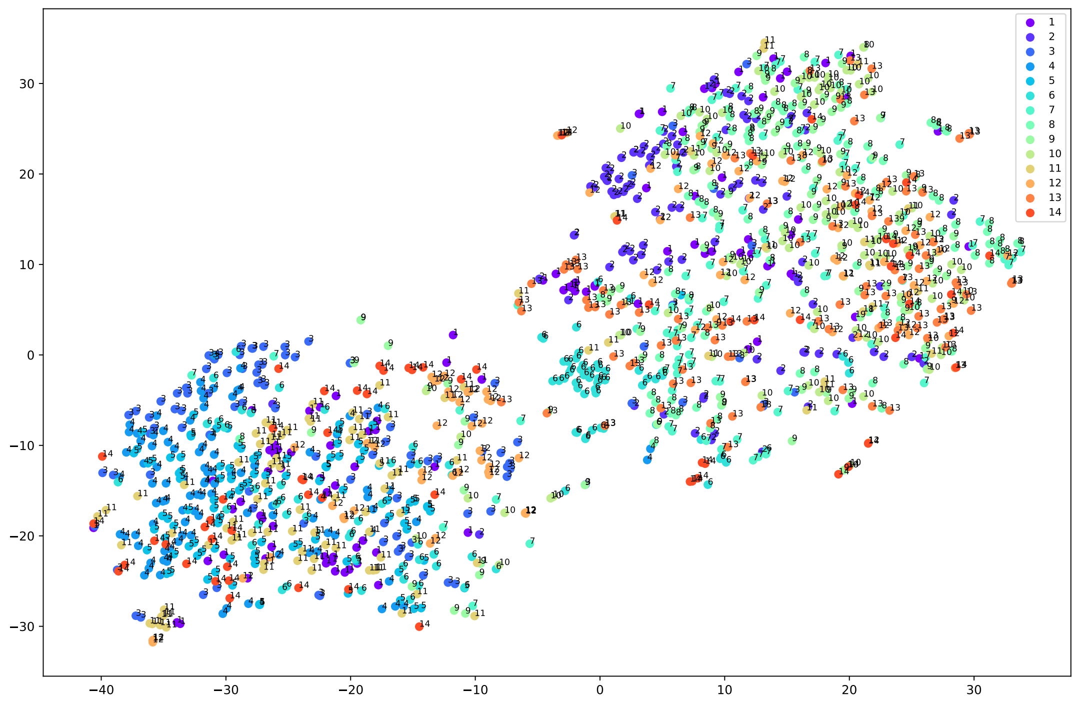
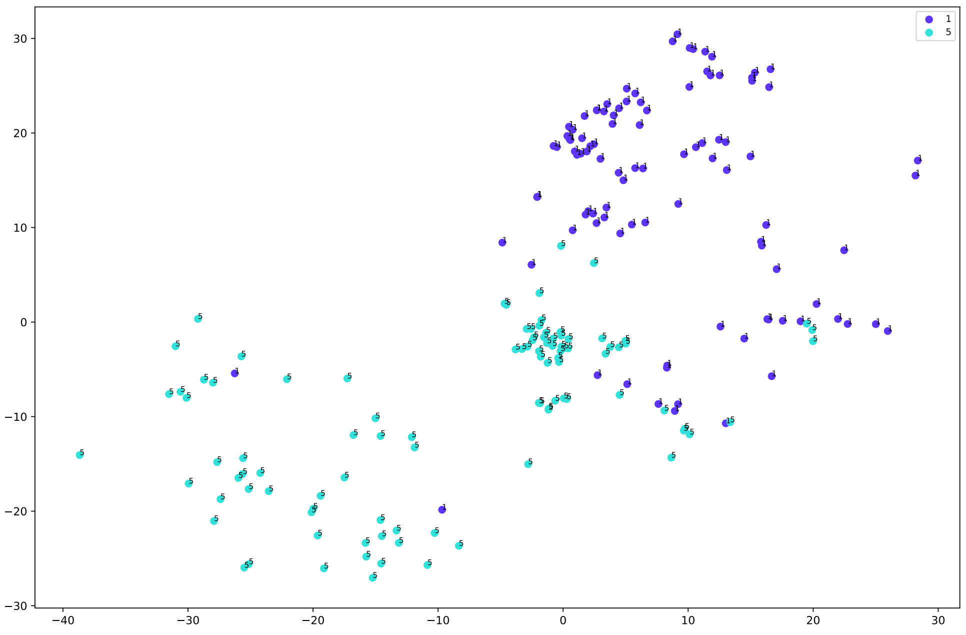


Figure 4.12: Visualization of extracted features from DeepLD for individual SNPs.


(a) Visualization of SNP features: bin 1 vs bin 5

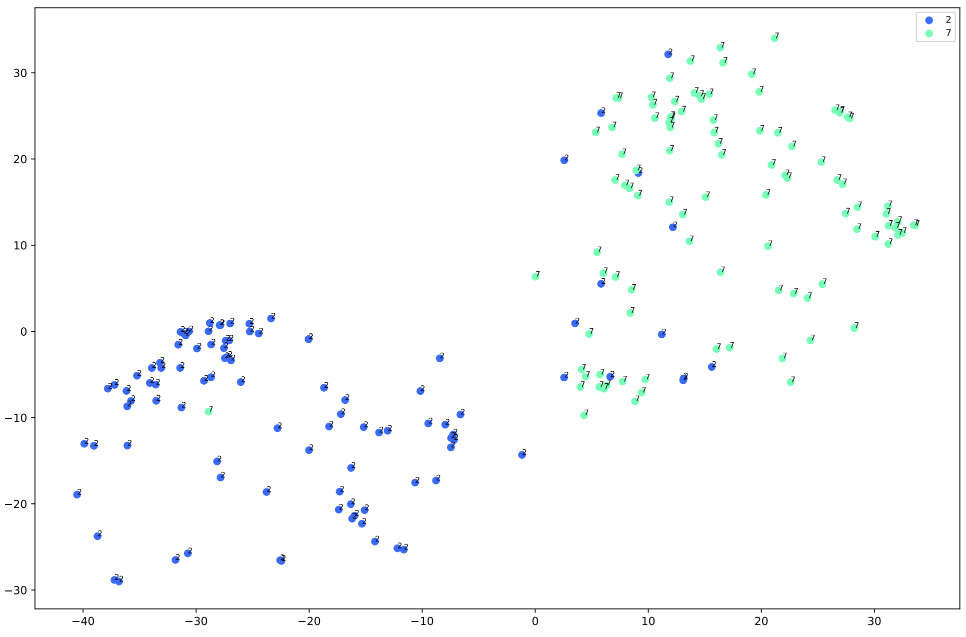

(b) Visualization of SNP features: bin 2 vs bin 7

Figure 4.13: Features learned for individual SNPs tend to localize in the latent space generated by t-SNE

4.10 Discussion

In this chapter I extended the model that we used in Chapter 3 to learn the relation between pairs of pairs of SNPs. As before, we used a Siamese based pattern of training our model to learn features that well represent each SNP, given its annotation. A distinction between the data that we dealt with in this chapter, compared to the previous one, is that the target that we are learning here is less erroneous and noisy as LD between pairs of variants can be readily computed upto a confidence level, if a moderately sized population with genomic data exists. Therefore, from that perspective, the only noise that might have been added to the network is due to the a wrong SNP calling which is negligible for the purpose of the task we are learning. Nonetheless, a challenge similar to the missing heritability still applies to the underlying task, that is, the SNP annotations can only be partially attributed to the loci LD structure. In fact many factors can impact the LD structure directly or indirectly. The most notable ones of which would be the natural selection [100], genetic drift [101], population subdivision/exchange and bottlenecks [102, 103], or even an interaction between them (e.g. an interaction between selection and genetic drift [104]).

4.11 Shortcomings and Future Works

While in this study, we computationally proved the existence of an association between variants annotations and LD structure of the corresponding loci harboring them, to best of our knowledge, there has not been any biological evidence proving this yet. In fact, this problem has never been brought to researchers attention, therefore, no literature exists on this potential research area, yet. Furthermore, we do not yet know how this association has been formed and whether or not this association is a cause-and-effect kind of relation. If yes, then which one caused the other? If no, then perhaps they are both derived from other causal factors that might also have to do with the factors that shape LD, as we listed above. These questions are remained to be answered by follow up studies and experiments.

While the main motivation for our study traces back to recent published works on improving GWAS power, the scope of our analysis in this chapter was limited to learning a relationship between genomic loci LD structures and SNPs annotations of the underlying variants, through learning a representation and by using a non-linear mapping. By all means, such a representation can be incorporated in GWA studies as new features, instead of placing a prior on the effect sizes. Such an approach, would of course increase the dimensionality of the input features, which needs to be addressed efficiently, to avoid the overfitting problem. Another potential direction to improve the model will the inclusion of recombination maps [105]. DeepLD as it stands now, segregates the contribution of variants distance to the LD scores, from the that of annotations. We however, do not include recombination maps into our model. This can result in learning spurious relations as locations that are active sites of recombination (such as recombination hotspots, in the extreme case) can cause an excessive reduction in the rate of LD between variants nearby. Fusing such information will help finding a more annotation-rich representation of variants. Fortunately, rich compendium of recombination maps for human genome are available and can be readily used to expand the model we proposed in this chapter.

With regard to the scalability, the implemented algorithm as it stands now is quite slow. This is for the most part due to the large number of variants triplets that should be fed to the model for training. More specifically, to properly train DeepLD, we need to present it with $O(K.N^3)$ training samples, where K is the number of loci that we are train on, and $N = \max_{k=1\ldots K}[N_k]$ with N_k being the number of variants in locus k. Such model is not feasible to train on the genome scale. Therefore, one of the key aspects to improve upon is to find faster ways of training, either by restricting the search space or by using a more efficient implementation of the method, or both.

CHAPTER 5
CONCLUSION

The research I pursued in my PhD work falls under two relevant tracks, a) characterization of transcriptional regulatory elements (Chapter 2), and b) modeling genome-wide associations and linkage disequilibrium using genomic and evolutionary annotation of variants (Chapters 3 and 4).

In the first track, I developed and benchmarked a series of models for prediction of protein binding preferences, starting from DeeperBind and ending to DeeperBind Plus. In doing so, I took advantage of advanced techniques that have been successfully employed by the machine learning community, in other domains, such as vision. I showed that in order to fully harness the power of genomic features and the underlying technology, one needs to directly model each aspect by including the right component in the model. Specifically, the aspects that we addressed extensively were, the overall temporal contribution of motifs and genomic features on probes, the potentially varying lengths of genomic features, the reverse complimentarity that is specific to the underlying technology, and the existence of noise in measurements. Moreover, I showed that by slightly changing the underlying problem from a regression model predicting normalized intensity into a classifier that learns to compare probes' corresponding binding affinities, we can gain more prediction power from such models.

The concrete goals of the first track of my dissertation were to develop a series of computational methods to unravel the patterns of transcriptional binding, using the DNA sequence only[1], as well as characterization and visualization of protein binding sites using the designed pipelines.

[1]While the methods I designed are equally applicable (with possibly minor changes,) to other data acquired by different high-throughput technologies, our main focus in this thesis has been on utilization of PBM.

The second track of my past research was devoted to developing methods for modeling genome-wide associations and the linkage disequilibrium. Both of these tasks pose similar challenges that restrict our ability in utilizing recent advances in deep learning research. Specifically, when dealing with GWA studies, we are often bound by high dimensionality of variants data, a significant degree of missing information (i.e. missing heritability), high complexity weak patterns to learn, and relatively small datasets. As a consequence, the state-of-the-art approaches for GWAS that are used in practice are different variations of linear models. In my thesis, I showed that part of the failure in learning higher-capacity models can be attributed to how we are training such models. Specifically, I showed that using Siamese networks and tools from graph theory we can achieve a performance higher or on par with the state-of-the-art Bayesian non-parametric approaches. Being successful in learning weak relationships using the proposed model, I then extended my approach to show that there is a relation between variants annotations and their underlying haplotype structure, which was not known before. Existence of such a relationship can increase the power of GWA models and if proved biologically will have important implications in population genetics.

The concrete goals of the second track are as follows. In Chapter 3 I show that instead of training a model that directly predicts phenotype values, it is more efficient to learn some noisy phenotype graph through a model that compares pairs of examples and subsequently propose an approach based on randomized optimization to infer a complete directed acyclic graph from the noisy phenotype graph, in case we need to rank samples. I Chapter 4 I show computationally, that pairwise SNP linkage disequilibriums are linked to functional annotation of the underlying SNPs. By developing a model that is capable of picking up weak LD relations from function annotations of SNP triplets, I verify that there is an interesting association between LD scores and annotations. To my knowledge, such an association has never been known or reported.

5.1 Concrete Innovation Deliverables

The key innovations in my dissertation, for each chapter, are as follows:

- (Chapter 2) uses recurrent neural networks to utilize the informative temporal features for more accurate identification of protein binding sites.
- (Chapter 2) encodes both sequences on the reverse complement strand as well as the direct strand in an efficient way that avoids redundant learning and overfitting.
- (Chapter 2) addresses varying length genomic features by directly modeling then using inception modules.
- (Chapter 2) proposes a new training strategy utilizing the Siamese networks architecture for training networks that are more robust to the noise in data and are significantly more accurate.
- (Chapter 2) proposes DeeperBind Plus, a hybrid approach that integrates all the above techniques into an end-to-end trainable architecture with a synergistic performance boost.
- (Chapter 3) shows that we can expect performances better or on par with the state-of-the-art non-parametric methods, by learning a phenotype graph instead of expression traits.
- (Chapter 3) proves that the inference of a denoised complete phenotype DAG is NP-complete, and provides a solution based on randomized optimization, to find a semi-optimal partial order.
- (Chapter 3) shows that using the above strategy for training, we can get the same or better performance from parametric methods, compared to the best non-parametric approaches.

- (Chapter 4) proposes a pipeline for capturing the relationship between functional annotation of variants and the LD score between them.
- (Chapter 4) successfully verifies the ability of this model to predict the LD between pairs of variants from unseen loci, using functional annotations variants only.

5.2 Concluding Remarks

In this dissertation, I elaborated a series of challenges for a few important problems in the Bioinformatics domain. I then addressed some of them and discussed and implemented methods to overcome the challenges, and reported improvements achieved by each, leaving the remaining challenges for future work. I have also alluded to a few directions where the unsolved challenges can be potentially solved or mitigated. Overall, my work lays the foundation for employing recent advanced machine learning ideas to solving long standing computational problems in Bioinformatics.

Appendices

APPENDIX A
SUMMARY OF THE PROOF OF THEOREM 4.3.1

In this appendix chapter, we summarize the first part of the proof of Theorem 4.3.1 from [92].

Let's assume that ϕ, an $N \times 1$ vector of phenotypes, is characterized by the following equation:

$$\phi = X\beta + \epsilon \tag{A.1}$$

where X is an $N \times M$ zero-mean unit-variance normalized genotypes, β is an $M \times 1$ vector of effect sizes and the residual term ϵ accounts for the contribution of the environment. Let's assume that all three variables in the right-hand side of equation A.1 are random variables with $\mathbb{E}[\epsilon] = 0$, $Var[\epsilon] = (1 - h_g^2)\mathbf{I}$, $\mathbb{E}[\beta] = \mathbf{0}$, and $Var[\beta] = (h_g^2/M)\mathbf{I}$. Let's assume an independence between the genotypes across individuals, and define the LD between genotypes j, k as $r_{jk} = \mathbb{E}[X_{ij}X_{ik}]$. Let's also assume a mutual independence between X, β, and ϵ. In this summary we only discuss the case where environmental effects are considered independent of genotypes[1].

For each variant j, $\hat{\beta}_j$, the least-square estimate of β_j, can be computed as $\hat{\beta}_j = X_{:j}^T\phi/N$. Furthermore, the chi^2-statistics can be computed as $\chi^2 = N\hat{\beta}_j^2$. The expectation of this statistics with respect to X, β, and ϵ will be computed as follows.

$$\mathbb{E}[\chi_j^2] \approx \frac{Nh_g^2}{M} l_j + 1 \tag{A.2}$$

where l_j, the LD score of SNP j, and is defined as $l_j = \sum_{k=1}^{M} r_{jk}^2$.

Observer that $\mathbb{E}[\chi_j^2] = N\mathbb{E}[\hat{\beta}_j^2] = N(\mathbb{E}[\hat{\beta}_j^2] - \mathbb{E}[\hat{\beta}_j]^2) = N.Var[\hat{\beta}_j]$ where the second equation is due to $\mathbb{E}[\hat{\beta}_j] = 0$. On the other hand, $Var[\hat{\beta}_j]$ can be computed using the law

[1]Interested reader can refer to the supplementary materials provided for [92].

of total variance, as follows:

$$Var[\hat{\beta}_j] = \mathbb{E}[Var[\hat{\beta}_j|X]] + Var[\mathbb{E}[\hat{\beta}_j|X]] = \mathbb{E}[Var[\hat{\beta}_j|X]] \tag{A.3}$$

Hence, we have,

$$\mathbb{E}[Var[\hat{\beta}_j|X]] = \frac{1}{N^2}Var[X_{:j}^T\phi|X] = \tag{A.4}$$

$$\frac{1}{N^2}X_{:j}^T Var[\phi|X]X_{:j} = \frac{1}{N^2}\left(\frac{h_g^2}{M}X_{:j}^T XX^T X_{:j} + N(1-h_g^2)\right) \tag{A.5}$$

Let's define the sample correlation between SNPs i, j as $\tilde{r}_{jk} = \frac{1}{N}\sum_{i=1}^{N} X_{ij}X_{ik}$, as a result, the term $\frac{1}{N^2}X_{:j}^T XX^T X_{:j}$ will be equal to $\sum_{k=1}^{M}\tilde{r}_{jk}^2$. Note that, $\mathbb{E}[\tilde{r}_{jk}^2] = r_{jk}^2 + (1-r_{jk}^2)/N$, hence,

$$\mathbb{E}\left[\sum_k \tilde{r}_{jk}^2\right] \approx l_j + \frac{M-l_j}{N} \tag{A.6}$$

and therefore,

$$\mathbb{E}[\chi_j^2] \approx \frac{N(1-1/N)h_g^2}{M}l_j + 1 \approx \frac{Nh_g^2}{M}l_j + 1 \tag{A.7}$$

Note that, we can relax the constraint regarding the population stratification in which case a term Na will be added to the above equation, where a measures the contribution of confounding factors [92].

APPENDIX B

LIST OF PROCESSED ANNOTATION FEATURES USED BY DEEPLD

a1:A	a1:C	a1:G
a1:T	a2:A	a2:C
a2:G	a2:T	CodingTranscript
Intergenic	NonCodingTranscript	RegulatoryFeature
Transcript	3PRIME_UTR	5PRIME_UTR
DOWNSTREAM	INTERGENIC	INTRONIC
NONCODING_CHANGE	NON_SYNONYMOUS	REGULATORY
SPLICE_SITE	SYNONYMOUS	UPSTREAM
ConsScore	3_prime_UTR	5_prime_UTR
NMD,3_prime_UTR	NMD,missense	downstream
intergenic	intron	intron,NMD
intron,non_coding	mature_miRNA	missense
non_coding_exon	regulatory	splice,5_prime_UTR
splice,NMD,missense	splice,intron	splice,intron,non_coding
splice,missense	splice,non_coding_exon	splice,synonymous
synonymous	upstream	GC
CpG	motifECount	motifEHIPos
motifEScoreChng	A	C
D	E	F
G	H	I
K	L	M
N	P	Q

R	S	T
V	Y	A
C	D	E
F	G	H
I	K	L
M	N	P
Q	R	S
T	V	W
Y	cDNApos	relcDNApos
CDSpos	relCDSpos	protPos
relProtPos	hmmpanther	lcompl
ncoils	ndomain	other
sigp	Dst2Splice	ACCEPTOR
DONOR	minDistTSS	minDistTSE
deleterious	tolerated	SIFTval
benign	possibly_damaging	probably_damaging
PolyPhenVal	priPhCons	mamPhCons
verPhCons	priPhyloP	mamPhyloP
verPhyloP	bStatistic	targetScan
mirSVR-Score	mirSVR-E	mirSVR-Aln
cHmm_E1	cHmm_E2	cHmm_E3
cHmm_E4	cHmm_E5	cHmm_E6
cHmm_E7	cHmm_E8	cHmm_E9
cHmm_E10	cHmm_E11	cHmm_E12
cHmm_E13	cHmm_E14	cHmm_E15
cHmm_E16	cHmm_E17	cHmm_E18

cHmm_E19	cHmm_E20	cHmm_E21
cHmm_E22	cHmm_E23	cHmm_E24
cHmm_E25	GerpRS	GerpRSpval
GerpN	GerpS	tOverlapMotifs
motifDist	EncodeH3K4me1-sum	EncodeH3K4me1-max
EncodeH3K4me2-sum	EncodeH3K4me2-max	EncodeH3K4me3-sum
EncodeH3K4me3-max	EncodeH3K9ac-sum	EncodeH3K9ac-max
EncodeH3K9me3-sum	EncodeH3K9me3-max	EncodeH3K27ac-sum
EncodeH3K27ac-max	EncodeH3K27me3-sum	EncodeH3K27me3-max
EncodeH3K36me3-sum	EncodeH3K36me3-max	EncodeH3K79me2-sum
EncodeH3K79me2-max	EncodeH4K20me1-sum	EncodeH4K20me1-max
EncodeH2AFZ-sum	EncodeH2AFZ-max	EncodeDNase-sum
EncodeDNase-max	EncodetotalRNA-sum	EncodetotalRNA-max
Grantham	Dist2Mutation	Freq100bp
Rare100bp	Sngl100bp	Freq1000bp
Rare1000bp	Sngl1000bp	Freq10000bp
Rare10000bp	Sngl10000bp	CTCF Binding Site
Enhancer	Open chromatin	Promoter
Promoter Flanking Region	TF binding site	dbscSNV-ada_score
RemapOverlapTF	RemapOverlapCL	RawScore
PHRED		

REFERENCES

[1] A Annunziato, “Dna packaging: Nucleosomes and chromatin,” *Nature Education*, vol. 1, no. 1, p. 26, 2008.

[2] F. H. Crick, “On protein synthesis,” in *Symp Soc Exp Biol*, vol. 12, 1958, p. 8.

[3] F. Crick, “Central dogma of molecular biology,” *Nature*, vol. 227, no. 5258, p. 561, 1970.

[4] S. P. B. A. G. M. L. R. L. James D. Watson Tania A. Baker, *Molecular Biology of the Gene (7th Edition)*.

[5] G. D. Stormo, “Modeling the specificity of protein-dna interactions,” *Quantitative biology*, vol. 1, no. 2, pp. 115–130, 2013.

[6] M. F. Berger, A. A. Philippakis, A. M. Qureshi, F. S. He, P. W. Estep III, and M. L. Bulyk, “Compact, universal dna microarrays to comprehensively determine transcription-factor binding site specificities,” *Nature biotechnology*, vol. 24, no. 11, p. 1429, 2006.

[7] H. Chial, “Mendelian genetics: Patterns of inheritance and single-gene disorders,” *Nature Education*, vol. 1, no. 1, p. 63, 2008.

[8] M. Slatkin, “Linkage disequilibrium—understanding the evolutionary past and mapping the medical future,” *Nature Reviews Genetics*, vol. 9, no. 6, p. 477, 2008.

[9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[12] G. E. Dahl, T. N. Sainath, and G. E. Hinton, “Improving deep neural networks for lvcsr using rectified linear units and dropout,” in *2013 IEEE international conference on acoustics, speech and signal processing*, IEEE, 2013, pp. 8609–8613.

[13] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, “The importance of skip connections in biomedical image segmentation,” in *Deep Learning and Data Labeling for Medical Applications*, Springer, 2016, pp. 179–187.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[15] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[16] R. J. Williams and D. Zipser, “A learning algorithm for continually running fully recurrent neural networks,” *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.

[17] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] S. A. Lambert, A. Jolma, L. F. Campitelli, P. K. Das, Y. Yin, M. Albu, X. Chen, J. Taipale, T. R. Hughes, and M. T. Weirauch, “The human transcription factors,” *Cell*, vol. 172, no. 4, pp. 650–665, 2018.

[19] H. R. Hassanzadeh and M. D. Wang, “Deeperbind: Enhancing prediction of sequence specificities of dna binding proteins,” in *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2016, pp. 178–183.

[20] T. Siggers and R. Gordaˆn, “Protein–dna binding: Complexities and multi-protein codes,” *Nucleic acids research*, vol. 42, no. 4, pp. 2099–2111, 2013.

[21] S. A. Jaeger, E. T. Chan, M. F. Berger, R. Stottmann, T. R. Hughes, and M. L. Bulyk, “Conservation and regulatory associations of a wide affinity range of mouse transcription factor binding sites,” *Genomics*, vol. 95, no. 4, pp. 185–195, 2010.

[22] R. Rohs, S. M. West, A. Sosinsky, P. Liu, R. S. Mann, and B. Honig, “The role of dna shape in protein–dna recognition,” *Nature*, vol. 461, no. 7268, p. 1248, 2009.

[23] Z Otwinowski, R. Schevitz, R.-G. Zhang, C. Lawson, A Joachimiak, R. Marmorstein, B. Luisi, and P. Sigler, “Crystal structure of trp represser/operator complex at atomic resolution,” *Nature*, vol. 335, no. 6188, p. 321, 1988.

[24] N. C. Seeman, J. M. Rosenberg, and A. Rich, “Sequence-specific recognition of double helical nucleic acids by proteins,” *Proceedings of the National Academy of Sciences*, vol. 73, no. 3, pp. 804–808, 1976.

[25] R. Gordân, N. Shen, I. Dror, T. Zhou, J. Horton, R. Rohs, and M. L. Bulyk, “Genomic regions flanking e-box binding sites influence dna binding specificity of bhlh

transcription factors through dna shape," *Cell reports*, vol. 3, no. 4, pp. 1093–1104, 2013.

[26] J. Kim and K. Struhi, "Determinants of half-site spacing preferences that distinguish ap-1 and atf/creb bzip domains," *Nucleic acids research*, vol. 23, no. 13, pp. 2531–2537, 1995.

[27] J. M. Franco-Zorrilla, I. López-Vidriero, J. L. Carrasco, M. Godoy, P. Vera, and R. Solano, "Dna-binding specificities of plant transcription factors and their potential to define target genes," *Proceedings of the National Academy of Sciences*, vol. 111, no. 6, pp. 2367–2372, 2014.

[28] M. T. Weirauch, A. Cote, R. Norel, M. Annala, Y. Zhao, T. R. Riley, J. Saez-Rodriguez, T. Cokelaer, A. Vedenko, S. Talukder, *et al.*, "Evaluation of methods for modeling transcription factor sequence specificity," *Nature biotechnology*, vol. 31, no. 2, pp. 126–134, 2013.

[29] M. Tompa, N. Li, T. L. Bailey, G. M. Church, B. De Moor, E. Eskin, A. V. Favorov, M. C. Frith, Y. Fu, W. J. Kent, V. J. Makeev, A. A. Mironov, W. S. Noble, G. Pavesi, G. Pesole, M. Régnier, N. Simonis, S. Sinha, G. Thijs, J. van Helden, M. Vandenbogaert, Z. Weng, C. Workman, C. Ye, and Z. Zhu, "Assessing computational tools for the discovery of transcription factor binding sites," *Nature Biotechnology*, vol. 23, no. 1, pp. 137–144, 2005.

[30] M. F. Berger, A. A. Philippakis, A. M. Qureshi, F. S. He, P. W. Estep, and M. L. Bulyk, "Compact, universal DNA microarrays to comprehensively determine transcription-factor binding site specificities.," *Nature biotechnology*, vol. 24, no. 11, pp. 1429–1435, 2006.

[31] X. Chen, T. R. Hughes, and Q. Morris, "RankMotif++: a motif-search algorithm that accounts for relative ranks of K-mers in binding transcription factors," *Bioinformatics*, vol. 23, no. 13, pp. i72–i79, 2007.

[32] A. Tanay, "Extensive low-affinity transcriptional interactions in the yeast genome," *Genome Research*, vol. 16, no. 8, pp. 962–972, 2006.

[33] K.-C. Wong, T.-M. Chan, C. Peng, Y. Li, and Z. Zhang, "DNA motif elucidation using belief propagation," *Nucleic Acids Research*, vol. 41, no. 16, e153, 2013.

[34] Y. Zhao and G. D. Stormo, "Quantitative analysis demonstrates most transcription factors require only simple models of specificity," *Nature biotechnology*, vol. 29, no. 6, pp. 480–483, 2011.

[35] Y. Orenstein and R. Shamir, “A comparative analysis of transcription factor binding models learned from pbm, ht-selex and chip data,” *Nucleic acids research*, vol. 42, no. 8, e63–e63, 2014.

[36] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, “Predicting the sequence specificities of dna-and rna-binding proteins by deep learning,” *Nature biotechnology*, vol. 33, no. 8, p. 831, 2015.

[37] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a” siamese” time delay neural network,” in *Advances in neural information processing systems*, 1994, pp. 737–744.

[38] G. Koch, R. Zemel, and R. Salakhutdinov, “Siamese neural networks for one-shot image recognition,” in *ICML deep learning workshop*, vol. 2, 2015.

[39] G. D. Forney, “The viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[40] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[41] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, “Tensorflow: A system for large-scale machine learning,” in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[42] C. Zhu, K. J. Byers, R. P. McCord, Z. Shi, M. F. Berger, D. E. Newburger, K. Saulrieta, Z. Smith, M. V. Shah, M. Radhakrishnan, *et al.*, “High-resolution dna-binding specificity analysis of yeast transcription factors,” *Genome research*, vol. 19, no. 4, pp. 556–566, 2009.

[43] D. E. Newburger and M. L. Bulyk, “Uniprobe: An online database of protein binding microarray data on protein–dna interactions,” *Nucleic acids research*, vol. 37, no. suppl_1, pp. D77–D82, 2008.

[44] M. L. Bulyk, *Uniprobe database*, 2006 (accessed July 14, 2019).

[45] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[46] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint arXiv:1312.6034*, 2013.

[47] G. E. Crooks, G. Hon, J.-M. Chandonia, and S. E. Brenner, "Weblogo: A sequence logo generator," *Genome research*, vol. 14, no. 6, pp. 1188–1190, 2004.

[48] J. Lanchantin, R. Singh, Z. Lin, and Y. Qi, "Deep motif: Visualizing genomic sequence classifications," *arXiv preprint arXiv:1605.01133*, 2016.

[49] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[50] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[51] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, 2013, p. 3.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[53] S. Chopra, R. Hadsell, Y. LeCun, *et al.*, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR (1)*, 2005, pp. 539–546.

[54] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[55] P. Zeng and X. Zhou, "Non-parametric genetic prediction of complex traits with latent dirichlet process regression models," *Nature communications*, vol. 8, no. 1, p. 456, 2017.

[56] N. Chatterjee, J. Shi, and M. García-Closas, "Developing and evaluating polygenic risk prediction models for stratified disease prevention," *Nature Reviews Genetics*, vol. 17, no. 7, p. 392, 2016.

[57] H. L. Allen, K. Estrada, G. Lettre, S. I. Berndt, M. N. Weedon, F. Rivadeneira, C. J. Willer, A. U. Jackson, S. Vedantam, S. Raychaudhuri, *et al.*, "Hundreds of variants clustered in genomic loci and biological pathways affect human height," *Nature*, vol. 467, no. 7317, p. 832, 2010.

[58] H. Li, Z. Peng, X. Yang, W. Wang, J. Fu, J. Wang, Y. Han, Y. Chai, T. Guo, N. Yang, *et al.*, "Genome-wide association study dissects the genetic architecture of oil biosynthesis in maize kernels," *Nature genetics*, vol. 45, no. 1, p. 43, 2013.

[59] M. C. Romay, M. J. Millard, J. C. Glaubitz, J. A. Peiffer, K. L. Swarts, T. M. Casstevens, R. J. Elshire, C. B. Acharya, S. E. Mitchell, S. A. Flint-Garcia, *et al.*, "Comprehensive genotyping of the usa national maize inbred seed bank," *Genome biology*, vol. 14, no. 6, R55, 2013.

[60] G. A. F. Júnior, G. J. Rosa, B. D. Valente, R. Carvalheiro, F. Baldi, D. A. Garcia, D. G. Gordo, R. Espigolan, L. Takada, R. L. Tonussi, *et al.*, "Genomic prediction of breeding values for carcass traits in nellore cattle," *Genetics Selection Evolution*, vol. 48, no. 1, p. 7, 2016.

[61] M. V. Rockman and L. Kruglyak, "Genetics of global gene expression," *Nature Reviews Genetics*, vol. 7, no. 11, p. 862, 2006.

[62] E. Grundberg, K. S. Small, Å. K. Hedman, A. C. Nica, A. Buil, S. Keildson, J. T. Bell, T.-P. Yang, E. Meduri, A. Barrett, *et al.*, "Mapping cis-and trans-regulatory effects across multiple tissues in twins," *Nature genetics*, vol. 44, no. 10, p. 1084, 2012.

[63] J. S. Hyams, S. D. Thomas, N. Gotman, Y. Haberman, R. Karns, M. Schirmer, A. Mo, D. R. Mack, B. Boyle, A. M. Griffiths, *et al.*, "Clinical and biological predictors of response to standardised paediatric colitis therapy (protect): A multicentre inception cohort study," *The Lancet*, vol. 393, no. 10182, pp. 1708–1720, 2019.

[64] U. of North Carolina. (2019). Predicting response to standardized pediatric colitis therapy, (visited on 09/09/2019).

[65] L. R. Lloyd-Jones, A. Holloway, A. McRae, J. Yang, K. Small, J. Zhao, B. Zeng, A. Bakshi, A. Metspalu, M. Dermitzakis, *et al.*, "The genetic architecture of gene expression in peripheral blood," *The American Journal of Human Genetics*, vol. 100, no. 2, pp. 228–237, 2017.

[66] U. of Queensland. (2017). Consortium for the architecture of gene expression, (visited on 09/09/2019).

[67] K. D. Joshi, *Foundations of discrete mathematics*. New Age International, 1989.

[68] T. M. Mitchell, *Machine learning*, 1997.

[69] P. Waldmann, G. Mészáros, B. Gredler, C. Fuerst, and J. Sölkner, "Evaluation of the lasso and the elastic net in genome-wide association studies," *Frontiers in genetics*, vol. 4, p. 270, 2013.

[70] O. Goldreich, *P, NP, and NP-Completeness: The basics of computational complexity*. Cambridge University Press, 2010.

[71] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*, Springer, 1972, pp. 85–103.

[72] M. Mooney, B. Wilmot, S. McWeeney, *et al.*, "The ga and the gwas: Using genetic algorithms to search for multilocus associations," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 3, pp. 899–910, 2012.

[73] J. rey Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Proceedings of the first IEEE conference on evolutionary computation, IEEE world congress on computational intelligence*, Citeseer, vol. 1, 1994, pp. 82–87.

[74] J. Yang, B. Benyamin, B. P. McEvoy, S. Gordon, A. K. Henders, D. R. Nyholt, P. A. Madden, A. C. Heath, N. G. Martin, G. W. Montgomery, *et al.*, "Common snps explain a large proportion of the heritability for human height," *Nature genetics*, vol. 42, no. 7, p. 565, 2010.

[75] G. Moser, S. H. Lee, B. J. Hayes, M. E. Goddard, N. R. Wray, and P. M. Visscher, "Simultaneous discovery, estimation and prediction analysis of complex traits using a bayesian mixture model," *PLoS genetics*, vol. 11, no. 4, e1004969, 2015.

[76] D. Speed and D. J. Balding, "Multiblup: Improved snp-based prediction for complex traits," *Genome research*, vol. 24, no. 9, pp. 1550–1557, 2014.

[77] S. H. Lee, J. H. van der Werf, B. J. Hayes, M. E. Goddard, and P. M. Visscher, "Predicting unobserved phenotypes for complex traits from whole-genome snp data," *PLoS genetics*, vol. 4, no. 10, e1000231, 2008.

[78] H. Kirsten, H. Al-Hasani, L. Holdt, A. Gross, F. Beutner, K. Krohn, K. Horn, P. Ahnert, R. Burkhardt, K. Reiche, *et al.*, "Dissecting the genetics of the human transcriptome identifies novel trait-related trans-eqtls and corroborates the regulatory relevance of non-protein coding loci," *Human molecular genetics*, vol. 24, no. 16, pp. 4746–4763, 2015.

[79] A. Majumdar, C. Giambartolomei, N. Cai, M. K. Freund, T. Haldar, T. Schwarz, J. Flint, and B. Pasaniuc, "Leveraging eqtls to identify individual-level tissue of interest for a complex trait," *bioRxiv*, p. 674 226, 2019.

[80] Q. Liao, Y. Ding, Z. L. Jiang, X. Wang, C. Zhang, and Q. Zhang, "Multi-task deep convolutional neural network for cancer diagnosis," *Neurocomputing*, vol. 348, pp. 66–73, 2019.

[81] W. Zhang, R. Li, T. Zeng, Q. Sun, S. Kumar, J. Ye, and S. Ji, "Deep model based transfer and multi-task learning for biological image analysis," *IEEE transactions on Big Data*, 2016.

[82] T. Zeng and S. Ji, "Deep convolutional neural networks for multi-instance multi-task learning," in *2015 IEEE International Conference on Data Mining*, IEEE, 2015, pp. 579–588.

[83] C. Sotiriou, S.-Y. Neo, L. M. McShane, E. L. Korn, P. M. Long, A. Jazaeri, P. Martiat, S. B. Fox, A. L. Harris, and E. T. Liu, "Breast cancer classification and prognosis based on gene expression profiles from a population-based study," *Proceedings of the National Academy of Sciences*, vol. 100, no. 18, pp. 10 393–10 398, 2003.

[84] S. Aubourg and P. Rouzé, "Genome annotation," *Plant Physiology and Biochemistry*, vol. 39, no. 3-4, pp. 181–193, 2001.

[85] N.-L. Sim, P. Kumar, J. Hu, S. Henikoff, G. Schneider, and P. C. Ng, "Sift web server: Predicting effects of amino acid substitutions on proteins," *Nucleic acids research*, vol. 40, no. W1, W452–W457, 2012.

[86] I. A. Adzhubei, S. Schmidt, L. Peshkin, V. E. Ramensky, A. Gerasimova, P. Bork, A. S. Kondrashov, and S. R. Sunyaev, "A method and server for predicting damaging missense mutations," *Nature methods*, vol. 7, no. 4, p. 248, 2010.

[87] P. Rentzsch, D. Witten, G. M. Cooper, J. Shendure, and M. Kircher, "Cadd: Predicting the deleteriousness of variants throughout the human genome," *Nucleic acids research*, vol. 47, no. D1, pp. D886–D894, 2018.

[88] M. J. Machiela and S. J. Chanock, "Ldlink: A web-based application for exploring population-specific haplotype structure and linking correlated alleles of possible functional variants," *Bioinformatics*, vol. 31, no. 21, pp. 3555–3557, 2015.

[89] G. Sveinbjornsson, A. Albrechtsen, F. Zink, S. A. Gudjonsson, A. Oddson, G. Másson, H. Holm, A. Kong, U. Thorsteinsdottir, P. Sulem, *et al.*, "Weighting sequence variants based on their annotation increases power of whole-genome association studies," *Nature genetics*, vol. 48, no. 3, p. 314, 2016.

[90] W. McLaren, B. Pritchard, D. Rios, Y. Chen, P. Flicek, and F. Cunningham, "Deriving the consequences of genomic variants with the ensembl api and snp effect predictor," *Bioinformatics*, vol. 26, no. 16, pp. 2069–2070, 2010.

[91] G. Kichaev, G. Bhatia, P.-R. Loh, S. Gazal, K. Burch, M. K. Freund, A. Schoech, B. Pasaniuc, and A. L. Price, "Leveraging polygenic functional enrichment to improve

gwas power," *The American Journal of Human Genetics*, vol. 104, no. 1, pp. 65–75, 2019.

[92] B. K. Bulik-Sullivan, P.-R. Loh, H. K. Finucane, S. Ripke, J. Yang, N. Patterson, M. J. Daly, A. L. Price, B. M. Neale, S. W. G. of the Psychiatric Genomics Consortium, *et al.*, "Ld score regression distinguishes confounding from polygenicity in genome-wide association studies," *Nature genetics*, vol. 47, no. 3, p. 291, 2015.

[93] H. K. Finucane, B. Bulik-Sullivan, A. Gusev, G. Trynka, Y. Reshef, P.-R. Loh, V. Anttila, H. Xu, C. Zang, K. Farh, *et al.*, "Partitioning heritability by functional annotation using genome-wide association summary statistics," *Nature genetics*, vol. 47, no. 11, p. 1228, 2015.

[94] B. Zeng, L. R. Lloyd-Jones, G. W. Montgomery, A. Metspalu, T. Esko, L. Franke, U. Vosa, A. Claringbould, K. L. Brigham, A. A. Quyyumi, *et al.*, "Comprehensive multiple eqtl detection and its application to gwas interpretation," *Genetics*, genetics–302 091, 2019.

[95] S. Purcell, B. Neale, K. Todd-Brown, L. Thomas, M. A. Ferreira, D. Bender, J. Maller, P. Sklar, P. I. De Bakker, M. J. Daly, *et al.*, "Plink: A tool set for whole-genome association and population-based linkage analyses," *The American journal of human genetics*, vol. 81, no. 3, pp. 559–575, 2007.

[96] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[97] T. Mitchell, *Machine learning*. McGraw-Hill Education, 1997.

[98] J.-H. Shin, S. Blay, B. McNeney, J. Graham, *et al.*, "Ldheatmap: An r function for graphical display of pairwise linkage disequilibria between single nucleotide polymorphisms," *Journal of Statistical Software*, vol. 16, no. 3, pp. 1–10, 2006.

[99] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[100] J. Felsenstein, "The effect of linkage on directional selection," *Genetics*, vol. 52, no. 2, p. 349, 1965.

[101] R. R. Hudson, "The sampling distribution of linkage disequilibrium under an infinite allele model without selection," *Genetics*, vol. 109, no. 3, pp. 611–631, 1985.

[102] M. Nei and W.-H. Li, "Linkage disequilibrium in subdivided populations," *Genetics*, vol. 75, no. 1, pp. 213–219, 1973.

[103] J. B. Mitton and R. K. Koehn, “Population genetics of marine pelecypods. iii. epistasis between functionally related isoenzymes of mytilus edulis,” *Genetics*, vol. 73, no. 3, pp. 487–496, 1973.

[104] W. G. Hill and A. Robertson, “The effect of linkage on limits to artificial selection,” *Genetics Research*, vol. 8, no. 3, pp. 269–294, 1966.

[105] F. Sun, M. Oliver-Bonet, T. Liehr, H. Starke, E. Ko, A. Rademaker, J. Navarro, J. Benet, and R. H. Martin, “Human male recombination maps for individual chromosomes,” *The American Journal of Human Genetics*, vol. 74, no. 3, pp. 521–531, 2004.