Regular and Phylogenetic Hidden Markov Models for Identifying Cell Type Specific Regulatory Regions

Navin Mordani

Master of Science

School of Computer Science

McGill University

Montreal,Quebec

2018-08-30

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Science

DEDICATION

This document is dedicated to all my teachers, the greatest of whom are my parents.

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude towards Professor Mathieu Blanchette, my thesis supervisor, without whose guidance, encouragement and insightful discussions the completion of this research work would not be possible. I thank him for showing faith in me and sponsoring my study. I am grateful to my parents and all other family members who have always supported and motivated me to settle for nothing less than excellence. Lastly, I would like to thank all my friends who made my graduate studies a memorable experience.

ABSTRACT

Transcriptional regulation of gene expression is a crucial process for the proper development and functioning of an organism. The process of transcriptional regulation ensuring the precise spatial and temporal expression of genes is carried out by proteins binding to regulatory regions in the DNA. Identification of these regulatory regions is crucial for understanding the patterns of gene expression regulation as well as for carrying out diagnosis and drug discovery for diseases occurring due to misregulation of genes caused by mutations in the regulatory regions. In this thesis, we present and compare computational methods based on regular hidden Markov models (HMMs), phylogenetic hidden Markov models (phylo-HMMs), and support vector machines (SVMs) for the identification of cell type specific regulatory regions using DNA sequence information from the genome of the species in question and from the genomes of related species. While the regular HMMs and SVMs only use DNA sequence information, the phylo-HMMs exploit sequence readout as well as sequence evolution information to identify regulatory regions. In the process of learning to identify regulatory regions, the models recognize highly discriminating sequence patterns indicative of regulatory function which are often either transcription factor binding motifs or portions of transcription factor binding motifs. Each of the three models has its shortcomings and on combining the predictions of two or more models; the models can complement each other to overcome these shortcomings. For instance, when training to identify GM12878 lymphoblastoid cell line specific regulatory regions in the human genome, we noticed that combining the predictions of the SVM, the regular HMM, and the phylo-HMM gave us the best results. Also, on comparing the highly discriminating

six-length sequence patterns recognized by the regular HMM and phylo-HMM against the database of known motifs we observed a total of 91 matches.

ABRÉGÉ

La régulation transcriptionnelle de l'expression des gènes est un processus crucial pour le bon développement et le bon fonctionnement d'un organisme. Le processus de régulation transcriptionnelle assurant l'expression spatiale et temporelle précise des gènes est réalisé par des protéines se liant aux régions régulatrices de l'ADN. L'identification de ces régions régulatrices est cruciale pour comprendre les schémas de régulation de l'expression génique, ainsi que pour effectuer le diagnostic et la découverte de médicaments pour les maladies dues à une mauvaise régulation des gènes provoqués par des mutations dans les régions régulatrices. Dans cette thèse, nous présentons et comparons des méthodes de calcul basées sur des modèles de Markov cachés réguliers (HMM), des modèles de Markov cachés phylogénétiques (phylo-HMM) et des machines à vecteurs de support (SVM) pour l'identification de régions régulatrices spécifiques de type cellulaire du génome de l'espèce en question et des génomes d'espèces apparentées. Alors que les HMM et SVM classiques n'utilisent que des informations sur les séquences d'ADN, les phylo-HMM exploitent les informations sur les séquences ainsi que sur l'évolution des séquences pour identifier les régions régulatrices. Dans le processus d'apprentissage pour identifier les régions régulatrices, les modèles reconnaissent des motifs de séquence hautement discriminants indiquant la fonction de régulation qui sont souvent soit des motifs de liaison de facteurs de transcription, soit des parties de motifs de liaison de facteurs de transcription. Chacun des trois modèles présente des lacunes et combine les prévisions de deux modèles ou plus, les modèles peuvent se compléter pour surmonter ces lacunes. Par exemple, lorsque nous nous sommes entraînés à identifier les régions régulatrices spécifiques

de la lignée cellulaire lymphoblastoïde GM12878 dans le génome humain, nous avons constaté que la combinaison des prédictions du SVM, du HMM normal et du phylo-HMM nous a donné les meilleurs résultats. En outre, en comparant les séquences de séquences de six longueurs hautement discriminantes reconnues par le HMM ordinaire et le phylo-HMM contre la base de données de motifs connus, nous avons observé un total de 91 correspondances.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
Introduction

1.1 Molecular Biology Background

DNA, or deoxyribonucleic acid, found inside the cell is a long double stranded molecule responsible for carrying genetic information. For simplicity it can be viewed as a string over the alphabet set {*A*(adenine), *C*(cytosine), *G*(guanine), *T*(thymine)}. Each element of this string is referred to as a *base* or *nucleotide*.

The information stored in the DNA contains the instructions needed for an organism to grow, survive and reproduce. In order to carry out these functions proteins need to be produced. A segment of the DNA that contains instructions to produce proteins is called as a *gene*.

1.1.1 Gene Expression

The process of utilizing the information carried in a gene to produce a protein is termed as *gene expression*. In eukaryotes gene expression involves 3 key steps, as shown in Figure 1–1:-

1. **Gene transcription**, in which the gene is transcribed to a precursor messenger RNA (precursor-mRNA).
2. **RNA splicing**, in which the introns are removed from the precursor-mRNA causing the precursor-mRNA to be converted to a mature messenger RNA. The gene architecture in eukaryotes (Figure 1–2) is divided into regions called exons which are interleaved with regions called introns. The introns are discarded from the gene

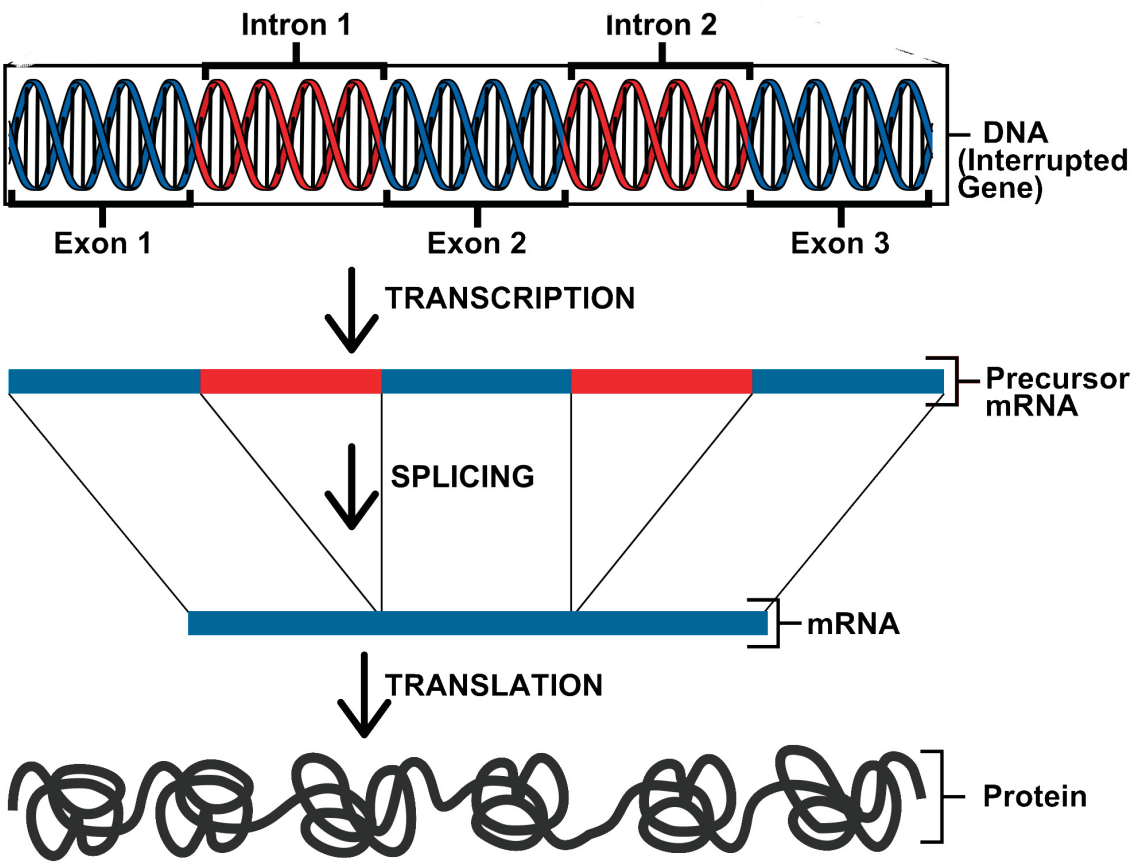

Figure 1–1: Gene Expression in Eukaryotes
(Figure taken from [2])

through splicing and are thus not translated into proteins. On the other hand the exons are translated into proteins. However, there are exons which are combination of translated (coding sequence (CDS)) and untranslated regions [1], which occur at the beginning and end of the gene called as 5' and 3' UTRs (untranslated regions).

3. **Translation**, in which the messenger RNA is translated to a protein.

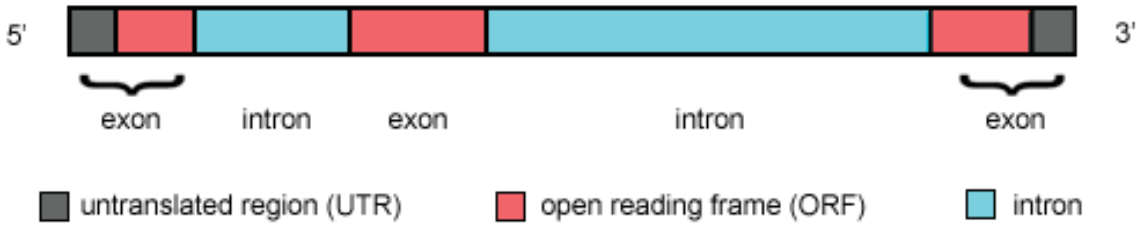

Figure 1–2: Eukaryotic Gene Architecture
(Figure taken from [3])

1.1.2 Need for Gene Expression Regulation

Gene expression regulation is the term used to describe any mechanism employed by the cell to enhance or repress the expression of a gene. The primary reasons for gene expression regulation are stated below.

Cell Differentiation

The cells forming different tissues of the human body have the same genome but have different functional roles. In order to fulfill these functions genes have to be selectively turned on and off. For example, genes encoding proteins needed for the function of the liver are turned on in the liver cells but are turned off in the heart cells.

Cell Division Control

Cyclin-dependent kinases (CDKs) are a family of protein kinases that are shown to have a role in regulating cell division [4]. The regulation of CDK encoding genes in turn leads to the regulation of cell division. Cell division is essential for long term tissue survival as cells have a limited lifetime. The rate of cell division needs to be regulated as different cells have different lifespans. For example, in humans red blood cells have an average lifespan of four months, while white blood cells live on average for more than a year [5].

Response to Environmental Changes

Gene expression is also altered in response to changes in the external environment or in immune response to a disorder. For example, it is shown that humans produce different amounts of skin pigment melanin depending on light exposure [6], and the gene producing the protein S100A9 which plays a prominent role in regulation of immune response [7] was recorded to be up-regulated in patients diagnosed with severe acute respiratory syndrome (SARS) [8].

1.1.3 Gene Transcription

Gene transcription is the first step in the process of gene expression. It is depicted in Figure 1–3 in which a gene is transcribed into precursor messenger RNA. This is the process that lies at the heart of the problem addressed in this thesis. In this process one of the DNA strands is used as a template strand to synthesize a complementary RNA (precursor messenger RNA) also called as the transcript of a gene. Transcription is done by an enzyme called RNA polymerase that binds to the promoter of the gene. The promoter is a fragment of DNA located upstream of the 5' end of the gene. Once the RNA polymerase is bound, the DNA double helix near the gene to be transcribed is unwound. The RNA polymerase then walks along the template strand synthesizing the complementary RNA by transcribing the complementary nucleotide to the RNA transcript for each nucleotide encountered along its walk. This synthesis originates from the transcription start site (TSS) which is the location where the first nucleotide is transcribed into RNA. The RNA polymerase continues adding nucleotides to the RNA strand until it gets signals to stop. The process of ending transcription is called termination, and it happens once the polymerase transcribes a sequence of DNA known as terminator.

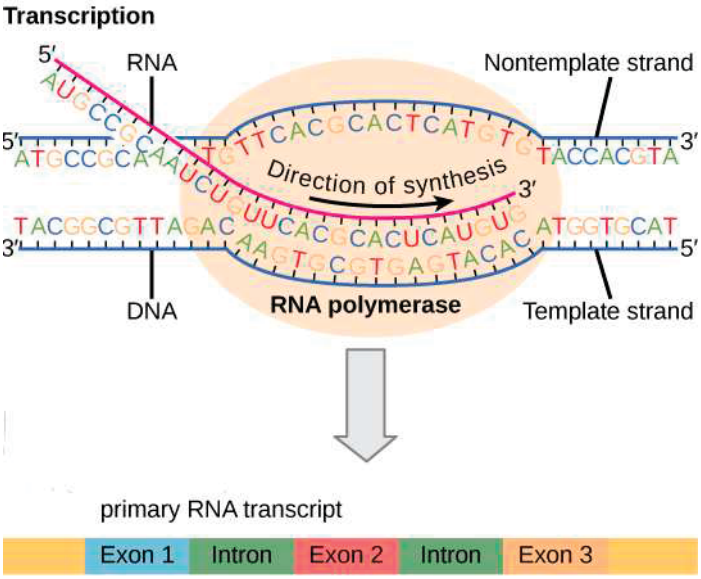

Figure 1–3: Gene Transcription
(Figure taken from [9])

1.1.4 Transcriptional Regulation

The RNA polymerase by itself is not equipped to recognize the TSS, and requires the presence of other proteins called transcription factors (TFs). The TFs bind the DNA at the promoter to form a complex thereby creating favourable conditions for the RNA polymerase to bind to the promoter forming another complex known as transcription initiation complex (TIC). There are several players regulating the formation of the TIC and hence regulating the transcription of the gene. These can be categorized into two broad sets: one, the TFs that bind the DNA, and the other that are a part of the DNA called as cis-regulatory regions or just regulatory regions.

1.1.5 Transcriptional Regulatory Factors

The rate of transcription is a result of the combined effect of gene activating and repressing mechanisms carried out by the regulatory factors.

Transcription Factors

Transcription factors (TFs) are proteins that control the rate of transcription of genes to precursor messenger RNA, by binding to specific DNA sequences. A census carried out on TFs in humans documents the presence of 1391 manually curated TFs and also speculates the presence of a total of 2000 - 3000 TFs [10].

The TFs bind DNA segments usually 5-15 bases long called *transcription factor binding sites* (TFBS). A set of TFBS for a TF is defined as a *motif*. The TFBS are a part of either the promoter region of a gene or enhancer regions, which are either in the introns or are further away from the TSS of the target gene than the promoters. They can be up to a distance of 1 million bases from the TSS either upstream or downstream of the target gene [11]. As mentioned earlier the TFs bind in a sequence-specific manner and a common way to represent binding specificities of a TF is by using a position weight matrix (PWM) [12] with each position in the binding site modelled as a multinomial distribution over the four nucleotides. A PWM for a particular TF is usually built by using SELEX [13] experiments.

The function of TFs is to regulate, i.e. turn on, and turn off genes to ensure that they are expressed in the right cells at the right time and in the right amount [14]. The TFs can be functionally classified as:

1. Activators: TFs that bind the DNA at promoters or at enhancers of a gene and activate transcription by attracting the TIC.

2. Repressors: TFs that inhibit the transcription of a gene. There are several ways in which this might be achieved. For example, the repressor may have the same binding preference as an activator and compete with it for binding sites. It is important to note that a repressor of one gene may act as an activator of the other gene.

There are also non-DNA binding proteins that assist the activators and repressors in gene expression regulation and are called as co-activators and co-repressors.

- Co-Activators: Non-DNA binding proteins, some of which act as a bridge between different activator proteins and help facilitate TIC formation while others help the activators to access their binding sites on the DNA.
- Co-Repressors: Non-DNA binding proteins that inhibit transcription. One of the common ways is to bind to an activator or to the TIC directly in order to inhibit their activity.

Transcription factors are modular in structure as can be seen from Figure 1–4 and these modules are called *protein domains*. A protein domain is a conserved part of a given protein sequence and structure that can evolve, function, and exist independently of the rest of the protein sequence. As described in [14], the protein domains that TFs contain are:

- DNA-binding domain (DBD), which attaches to specific sequences of DNA either in the promoter or enhancer regions of the regulated genes.
- Trans-activating domain (TAD), which contains binding sites for other proteins such as transcription co-activators and co-repressors. These binding sites are referred to as activation functions (AFs).

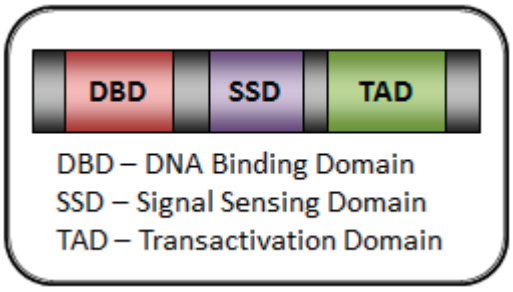

Figure 1–4: Structure of a TF
(Figure taken from [15])

- An optional signal-sensing domain (SSD), which senses external signals and, in response, transmits these signals to the rest of the transcription complex, resulting in up or down-regulation of gene expression. Also, the DBD and signal-sensing domains may reside on separate proteins that associate within the transcription complex to regulate gene expression.

Cis-Regulatory Regions

Cis-regulatory regions are portions of the DNA that harbour activators or repressors. They can be divided into the following classes:

1. Promoters: Promoters are found upstream of 5' end of the gene and are limited to a few hundred bases. These contain binding sites for the TFs that develop favourable conditions for binding of the RNA polymerase and also contain binding sites for other activators.
2. Enhancers: Enhancers are regions that are about 50 to 1500 base pairs in length containing binding sites for activators and are far away from the TSS of the target gene than the promoters. Their location with respect to the target gene (or genes) is highly variable. The identification of enhancers has been more challenging than

that of promoters. To identify promoters, regions upstream of the 5' end of the genes upto a few hundred bases need to be scanned instead of the entire genome. Also, the enhancers are not necessarily associated with the regulation of the closest gene and in some cases have been found to affect the regulation of multiple genes. Similarly, it is not uncommon for a gene to rely on contributions from multiple enhancers for its spatial and temporal regulation [16]. In fact mammalian genomes contain around 20,000 genes and an estimate of approximately 1 million enhancer sequences bringing the calculation to 4 enhancers per gene per cell type [17]. The mechanism that best explains how enhancers affect gene expression even from a distance of up to a million bases is that of enhancer-promoter looping, which is depicted in Figure 1–5. The DNA sequence in the cells, though often depicted as a linear sequence of nucleotides for simplicity is actually packaged tightly as a ball of yarn around nuclear proteins. The complex of DNA and nuclear proteins is termed as *chromatin*. Chromatin is a flexible polymer that can reorganize to form loops leading to enhancer promoter interaction in the right tissue at the right time [18]. The transcription factors bound to the promoters and enhancers interact with each other directly or through a co-activator. This increases the local concentration of transcription factors in the vicinity of the gene enhancing its expression [11].

3. Silencers: As the name suggests silencers are regulatory regions that have an inhibiting effect on transcription. Like enhancers these may be located thousands of bases away from the target gene, and contain binding sites for repressors.

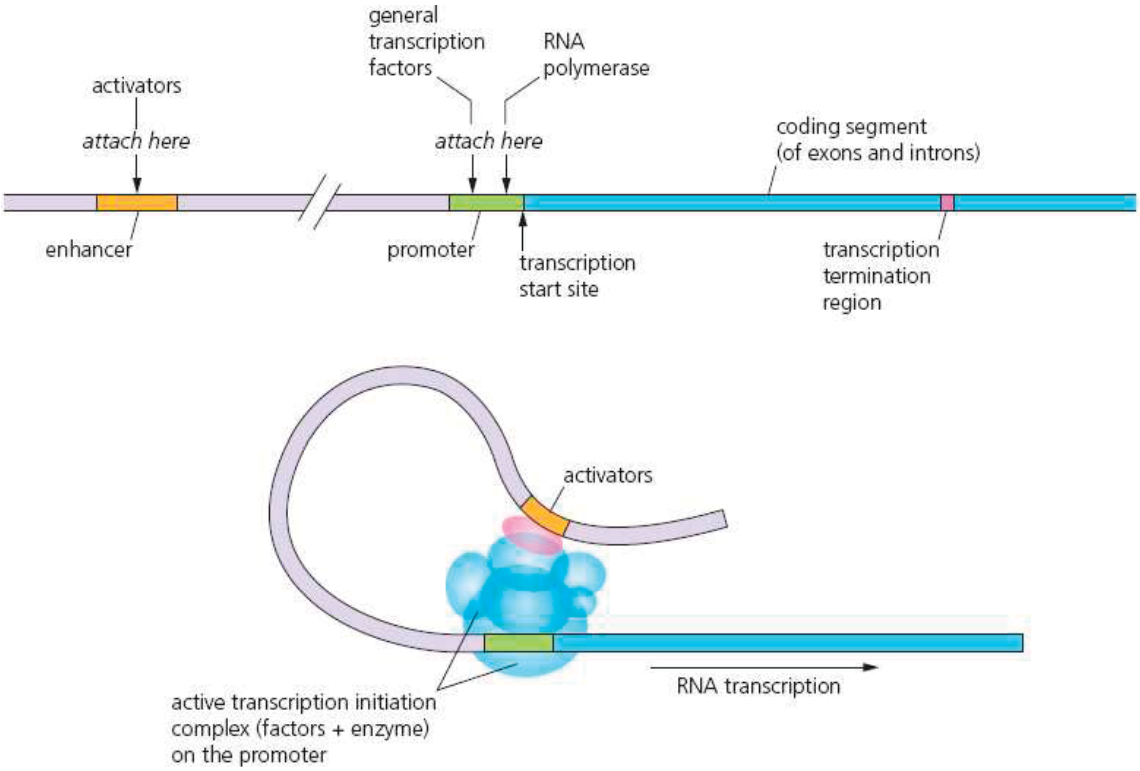

Figure 1–5: Enhancer-Promoter loop
The top portion shows the DNA in a linear fashion showing different components such as enhancers, promoters and genes. The lower portion of the figure shows the enhancer-promoter interaction due to looping. Different colours are used to make the mapping between the top and the bottom portion more clear (Figure taken from [19]).

1.1.6 Tissue Specificity of Regulatory Regions

As mentioned earlier in section 1.1.2, a cell has only a limited subset of its genes expressed. This subset depends on the cell type. This occurs owing to the fact that different tissues have different functional roles and to fulfill these functions genes need to be selectively expressed. As TFs are involved in the regulation of genes differential cell type specific gene expression is carried out by regulating selective binding of TFs to the DNA. The selective binding of TFs leads to only a subset of the cis-regulatory regions being functionally active. Hence, in a given cell type only a subset of the cis-regulatory regions are active. Based on a comparative study of gene expression, TF binding, and chromatin structure has revealed that chromatin structure plays the major role in differential binding of TFs [20]. A *nucleosome* is a subunit of chromatin where the DNA is wound around a set of eight proteins called histones. Each nucleosome takes up less than two turns of the DNA comprising of around 146 bases [21]. In a simplified way a nucleosome can be thought of as a cylinder with the DNA wound around it like a string. The DNA that is in the nucleosomes is generally inaccessible for the TFs to engage, making the cis-regulatory regions that fall in the nucleosomes inactive. What leads to cell-specific gene expression is the organization of the chromatin in different cells. The active re-arrangement of nucleosomes causing chromatin reorganization is carried out by *pioneering factors* which are special TFs that have the ability to bind nucleosomal DNA. Figure 1–6 shows the pioneering factor binding the nucleosomal DNA and opening up the chromatin, making it accessible to the other TFs.

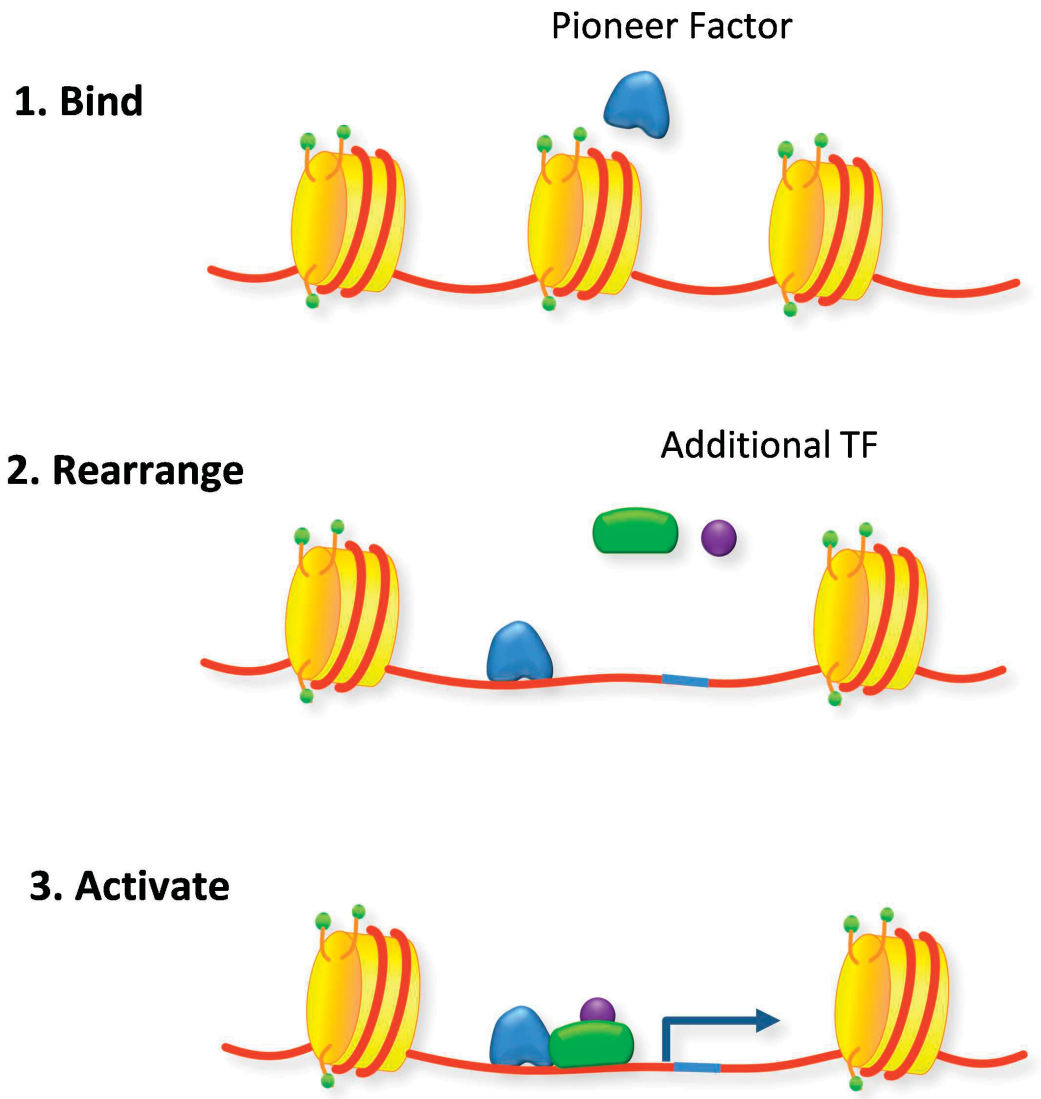

Figure 1–6: Opening up of Chromatin by Pioneer Factor
(Figure taken from [22]).

1.1.7 Sequence Conservation

All present day organisms have descended from a single common ancestor and their genetic sequences have evolved from ancestral sequences. The species are different due to the mutations accumulated in their DNA sequences. The mutations include deletions (removal of one or more bases from the DNA sequence), substitutions (replacement of a base by another base), and insertions (one or more bases added to an existing DNA sequence). Mutations may have positive effects or negative effects in the form of increase or decrease in the survival and reproduction rate, or be neutral. Selection is the powerful mechanism in evolution through which the variations produced by the mutations get filtered. The increase in survival and reproduction rates associated with positive changes enable the individuals having them to have a higher number of offsprings, in turn, causing these positive changes to be preserved, while the opposite happens in case of the negative changes and they are eventually removed from the population. Mutations in the functional regions such as genes and regulatory regions more often than not have negative impact, putting these mutations under negative selection whereas on the other hand neutral mutations accumulate in the non-functional regions. As a result the sequences of the non-functional regions become increasingly dissimilar as the evolutionary distance grows. Genes on the other hand are highly conserved amongst the species and hence many gene finding techniques have taken the approach of comparing orthologous sequences among multiple species [23]. Regulatory regions are a little tricky to deal with when it comes to sequence conservation mainly due to three reasons. Firstly, the effect of mutations occurring in regulatory regions may vary, some of the mutations may only have a subtle effect such as change in the gene expression levels while others may disrupt the function of the

region by changing or deleting a TFBS resulting in variations that are deleterious making them subject to negative selection. Secondly, sequence conservation may not be a must for the conservation of transcriptional regulation as TFs exhibit some flexibility in the underlying sequence of the sites they bind, and sometimes changes in the relative ordering and spatial relationship of binding sites do not result in loss of functionality [24, 25]. Even though it has been shown that sequences that are conserved over large evolutionary distances are more likely to be functional than those which are conserved over short distances [26], sequence conservation does not guarantee that the sequence is functional.

1.2 Biomedical Importance of Regulatory Regions

This thesis addresses the problem of identifying regulatory regions in the human genome. The problem is worthy enough for the purpose of disease diagnosis and drug discovery as many findings have shown a link between mutations in the regulatory regions in humans and many diseases [25]. These diseases may be hereditary or non-hereditary in nature depending on whether the causal mutation in the regulatory region belonged to the type *germline* or *somatic*. Germline mutations are mutations that may be transmitted to an offspring as they occur in the *germ cells*. Germ cells are cells that become sex cells also known as gametes. An instance of a hereditary disease shown to be linked with germline mutations in the regulatory regions is β-thalassemia, an inherited blood disorder that reduces the production of hemoglogin [27]. β-thalassemia is shown to be linked with mutations in the promoter of the β-globin gene causing disturbance in its expression [28]. Another hereditary disease shown to be linked with germline mutations is pyruvate kinase deficiency, an inherited disorder characterized by lack of the pyruvate kinase enzyme which is used by red blood cells. Due to the lack of pyruvate kinase enzyme the red blood

cells break down easily, leading to lower populations of these cells [29]. Pyruvate kinase deficiency is shown to be linked with mutations in the promoter of the PKLR gene [25]. On the other hand several non-hereditary diseases have been linked to somatic mutations in the regulatory elements. Somatic mutations unlike germline mutations do not occur in the germ cells and are not propagated to an offspring. Mutations in the promoter of the erythropoietin gene is shown to cause severe diabetic eye and kidney complications [30].

1.3 Experimental Approaches to Identify Regulatory Regions

There are several experimental approaches developed for identifying regulatory regions. Here, we discuss the two major high-throughput genome-wide approaches.

1.3.1 High-Throughput ChIP Experiments

Chromatin Immunoprecipitation (ChIP) [31] experiment is an experiment for probing protein-DNA interactions within the cell. It can be used to identify multiple genomic regions associated with a protein. The procedure for the ChIP experiment is as follows:-

1. The DNA and the interacting protein are cross-linked using a reversible cross-linking agent such as formaldehyde. This preserves the DNA-protein interactions.
2. The DNA-protein complexes are fragmented into fragments of approximately 500 bases by sonication.
3. The fragments of chromatin are then immunoprecipitated using antibodies specific to the particular protein. This will cause the fragments associated with proteins to be co-precipitated.
4. The cross-links between the protein and the DNA are reversed to get the DNA that was interacting with the protein.

The sequence of the DNA segments that were interacting with the proteins can be determined by either using microarrays (ChIP-chip) [32] or by using next-generation massively parallel sequencing (ChIP-Seq) [33, 34]. Both ChIP-chip and ChIP-seq experiments have been used to identify DNA regions in different organisms and cell types that are bound by several TFs on a genome wide basis. As the regulatory regions contain binding sites for multiple TFs, the aggregation of results of ChIP-chip and ChIP-seq experiments conducted for several TFs on a cell type is used as a representation of the regulatory regions in that cell type.

1.3.2 Assays Identifying Open Chromatin

The portions of the DNA present in the nucleosomes do not harbour binding sites for TFs and hence detecting the nucleosome free regions of the chromatin, also called open chromatin, can be useful for identifying regulatory regions. These assays are useful in scenarios when the DNA binding proteins are not known or the antibody is not available. DNASE-seq is one such approach. In DNASE-seq an enzyme called deoxyribonuclease (DNASE) is used to digest the open DNA, also called as DNASE-I hypersensitive sites. The mapping of the DNASE-I hypersensitive sites across the entire genome is done using next generation high-throughput sequencing technologies such as Illumina or Solexa [35, 36].

The experimental approaches discussed in this section have limitations in the form of the cost of the experiments, the number of TFs whose motifs can be profiled, and the number of different cell types for which the experiments can be carried out. Due to these limitations computational approaches towards handling the task of identifying cis-regulatory regions are an invaluable asset to the researchers.

1.4 Literature Review

The problem of computationally identifying regulatory regions has been approached in many different ways in the past. This section summarizes these approaches. The different approaches can be categorized based on the pieces of information they leverage to handle the task of identifying regulatory regions.

1.4.1 Using Densities of Transcription Factor Binding Sites

The computational tools in this category view regulatory regions as clusters of TFBS and use the already profiled motifs of TFs to identify clusters of TFBS in the DNA sequences. The crux of the methodology of these tools can be explained in two steps. Firstly, the DNA sequence is scanned for matches to PWMs of multiple TFs to find putative TFBS, and a score is assigned to the regions within the sequence based on a statistical measure of the significance of density of TFBS. Secondly, an attempt to filter out the regions that are spurious hits that occur just by random chance is made by setting a threshold upon the calculated score [25]. MCAST, a tool presented in [37] identifies regions with clusters of binding sites in a given DNA sequence by analyzing a window within the sequence whose length is specified by the user. The window is slid over the DNA sequence by a user-specified step-size. The putative TFBS or hits within the window are identified using the PWMs for multiple TFs. MCAST scores a window having multiple hits by undertaking the following steps. Firstly, it calculates the P-values for each hit for individual TF based on the local nucleotide distribution of the sequence within the window. The P-value of an individual binding site is equal to the expected frequency of equivalent or better sites in a random sequence having a similar nucleotide composition as that of the sequence in the window. Secondly, combined single hit P-values for multiple TFs are approximated

as upper bounds. Thirdly, scores for all k-hits within the window are calculated. A k-hit is defined as a set of non-overlapping k hits within a window, where k is a user-defined parameter signifying the least number of hits required for the window to be a contender for a regulatory module. The score of a k-hit is defined as the maximum of the P-values $p_1, p_2, ..., p_k$ of the k non-overlapping hits as the requirement is that all the P-values should be small. The k-hit score for the entire window is taken as the minimum of all the k-hit scores for that window. Finally, if the k-hit score of the window is below the decided threshold, then the window is reported as a regulatory region.

The work presented by Lifanov *et al.* in [38] which aimed at predicting regulatory regions associated with the developmental genes in the Drosophila genome involved the identification of homotypic clusters of binding sites, i.e. binding sites for a single TF as opposed to finding heterotypic clusters of binding sites from multiple TFs. This choice relied on the results presented in the earlier works [39]. The data required for building the classifier included known TF binding motifs to construct the PWMs, set of known cis-regulatory regions, and known regulatory interactions. The methodology comprised of counting the number of binding sites in a sliding window over the genome and evaluating the significance of the clusters in the form of an E-value by adopting a view that the number of PWM matches in a window follows a Poisson distribution [40].

Blanchette *et al.* [41] leveraged the fact of sequence conservation and shifted the focus from detecting regulatory regions associated with a particular set of genes to predicting a global map of regulatory regions for the entire human genome. The data used to fulfill this task included a total of 481 PWMs representing the binding affinities of a total of 229 TF families from the TRANSFAC database [42], and the genome-wide alignment

of the human, mouse, and rat genomes produced using the MULTIZ program. The regions from all the three genomes within the MULTIZ alignment blocks were scanned for PWM matches against all the PWMs, and the log-likelihood ratio scores were calculated, then for every alignment column, the hit scores across the three genomes were combined as a weighted sum. The human genome being the one whose regulatory map was to be uncovered was understandably given more weight than the hit scores of the positions from the other two genomes. The alignment columns with hit scores less than a threshold of 10 were discarded and then the scores for the regions were calculated. The score for a region lying a between alignment columns p_1 and p_2 under the PWM m was taken as the sum of the hit scores of the non-overlapping hits occurring within the region. Following this, P-values were calculated for the score of every region under every PWM, also taking into consideration the GC content and the length of the region. Finally, the candidate regulatory modules were viewed as consisting of hits from a range of one to five PWMs of the 481 PWMs and were assigned module scores by taking the negative of the log of the P-value for the event that a module consists of hits from k PWMs, where k lies within the range from one to five. The number k for a region was selected to be the one which recorded the highest statistical significance.

1.4.2 Using Combinatorial Effects of Transcription Factors

As discussed in section 1.1.5 gene expression regulation is a result of co-operative and competitive binding of TFs and hence identifying regulatory regions as by treating them as combinatorial patterns of TFBS has been a widely adopted approach. Wasserman and Fickett [43] employed logistic regression analysis (LRA) to predict skeletal muscle-specific regulatory regions in humans. A set of TFs regulating the gene expression in the

skeletal muscle cells was recognized, and their PWM match scores were used as features to train the LRA model. The same approach was later used to identify liver-specific regulatory regions with an additional step of filtering aimed at reducing the number of false positive predictions and improving specificity [44]. The filtering step involved the selection of predicted regions conserved between orthologous human and rodent sequences.

A large portion of this category is dominated by a family of probabilistic methods, all of which implement a hidden Markov model (HMM) to identify regulatory regions. They model the regulatory regions as being generated by a combination of binding sites. One of the first members of this family is the tool Cister [45] which uses posterior decoding to find for every position in a queried DNA sequence, the likelihood of being in the 'motif', 'inter-cluster background', and 'intra-cluster background' states. The emission probabilities for the motif states are evaluated from the nucleotide frequency matrices of TFs provided as input, whereas for the background states the emission probability values are estimated at runtime based on the frequency of the nucleotides found in a window surrounding the position in question. Cister was applied on the muscle data and was found to perform at least as well as the LRA method of Wasserman and Fickett [45].

Cluster-Buster [46] learns two separate HMMs given a set of frequency matrices for the TFs, and a queried sequence. One of them known as the motif cluster model, models the regulatory regions, and the other models the background DNA. To decide whether a subsequence within a queried sequence is a regulatory region or not, the likelihood of the subsequence as per both models is compared using the log odds ratio. The motif cluster model assumes that in a regulatory region the motifs occur randomly following a uniform distribution, whereas the background model is based on the assumption that nucleotides

occur randomly and independently with probabilities estimated from their frequency of occurrences in the neighbourhood of the queried position in the sequence. Cluster-Buster locates regulatory regions in a given input sequence by following a three-step algorithm described in [46].

Stubb [47] learns an HMM using the expectation-maximization technique to detect regulatory regions given the PWMs of a set of TFs. Along with the usual parameters of an HMM, Stubb also learns correlations between binding sites, and it does so by learning a history-conscious HMM. The choice for this variant was guided by the knowledge that some TFs in order to be functional need to interact with other TFs with the help of co-factors, and there were instances observed when modules predicted by the computational methods available then were reported as non-functional, probably because something in the arrangement of the binding sites was not proper. The process at every step either emits a motif of one of the TFs or emits the background motif. The probability of a motif also known as subsequence probability is calculated by using the entries of the weight matrices and is an important part of the computation required for training the HMM. StubbMS is an extension applied to Stubb. It compares sequences from multiple species to improve the detections of regulatory regions. StubbMS finds fixed alignment between species based on sequence similarity using Lagan [48] or DiAlign [49] depending on whether two or more than two species are used. The rest of the procedure for training the HMM remains the same except for the calculation of the subsequence probabilities for the motifs falling within the aligned blocks, which are calculated based on an evolutionary model that assumes that all the bases evolve independently at equal rates, and that the probability of fixation of a mutation is proportional to the values present in the PWMs.

Sinha *et al.* presented MORPHMS [50] that is similar to their previously designed tool StubbMS [47] but overcomes one shortcoming of StubbMS, and consequently performs better. StubbMS uses LAGAN [48] to compute the alignment between two species and assumes this alignment to be correct, whereas MORPHMS aligns the DNA sequences using a probabilistic approach. A pair-HMM similar to the one used by Holmes and Durbin [51] is used to compute the alignment. This allows the uncertainty in the alignment to be accounted for.

EEL [52] is a computational tool which was developed with the same agenda as MORPHMS, to combine both sequence alignment and detection of cis-regulatory regions. EEL first scans a pair of given orthologous sequences to locate all the putative binding sites and then uses the Smith Watermann local alignment algorithm [53] to align the binding sites. The scores assigned to the regions are based on binding site clustering, affinity of binding sites, and conservation. To account for the clustering factor, a negative score is added to the total score for the increased distance between adjacent binding sites and a positive score is added for the conserved binding sites based on their binding affinities. In order to account for the fact that true affinities of binding sites are not only dependent on the underlying sequence but also on the secondary interactions between TFs required for their cooperative binding, EEL adds correction factors which approximate the maximum free energy lost due to the loss of interactions between TFs caused by insertion of sequences between adjacent binding sites. This loss in free energy is approximated as the energy required to twist or compress DNA sequences of different lengths such that similar 3-D positions for both the pairs of TFs is achieved.

1.4.3 Motif Blind Techniques Using Sequence

Motif blind techniques do not use motifs of TFs to detect regulatory regions. These techniques use the underlying DNA sequence information to identify regulatory regions with some of the techniques also using other information such as sequence conservation. In the process of identifying discriminatory sequence patterns, these techniques often end up discovering motifs or portions of motifs of TFs. This is commonly named as the De novo discovery of binding sites. Elnitski *et al.* in their work presented in [54] computed Regulatory potential (RP) scores of regions to discriminate regulatory DNA from neutral DNA. The computation of RP scores takes into account both the underlying DNA sequence information as well as sequence conservation into account. Two different Markov models are learned, one to model the regulatory DNA, and the other to model the neutral DNA. These Markov models run along an alignment rather than along a single DNA sequence and the RP score for a region is given by the log-odds ratio from the learned Markov models. However, choosing the parameters of the Markov models is tricky especially with the increase in the number of species in the alignment. In [54] a fifth order Markov model along with a 5-symbol alphabet (matches of As and Ts, matches of Cs and Gs, substitutions comprising A and G or C and T called transitions, substitutions comprising A and T or C and G called transitions, and columns containing a gap) recorded the best performance on the human-mouse alignment, whereas in [55] for the three-way human, mouse, and rat alignment, a second order model using a more complicated 10-symbol alphabet performed the best.

CisModule [56] presented by Zhou and Wong aims at the detection of TFBS, motif patterns, and cis-regulatory regions by using hierarchical mixture modeling with Bayesian

inference. It uses a two-level hierarchical mixture model. At the first level, the sequences are modeled as a collection of regulatory regions of a user-specified length, interspersed with background sequences while the second level views the regulatory regions as a mixture of motifs, and inter-regulatory background regions.

HexDiff [57] counts the frequency of hexamers within a known set of regulatory regions, and a set of control sequences provided as training data. It then finds the most discriminating hexamers and these selected hexamers are then used for building a linear model to predict regulatory regions in new sequences. The idea behind this which was also validated by the authors was that the highly discriminating hexamers are mostly parts of sequence signatures of the binding sites found in the regulatory regions.

Kmer-SVM [58] uses support vector machine (SVM) classifier to classify DNA sequences. It trains an SVM classifier using a set of known regulatory regions as the positive set, and a negative set containing sequences that match the positive sequences in terms of GC dinucleotide content, length, and repeat fraction. The input features to the SVM classifier are a set of k-mers. As a result of the training process, weights are assigned to the k-mers signifying their power in discriminating between the sequences from the positive and the negative set. It offers a choice of two kernels to compute the similarity among sequences required by the SVM classifier: the spectrum kernel, and the weighted spectrum kernel.

1.4.4 Integrative Approaches

The computational tools falling under this category use a combination of more than one of the previously defined approaches. The computational tool CLARE [59] uses both, a set of known TF PWMs from the TRANSFAC [42] and JASPAR [60] databases as well

as de novo binding sites. The input to the CLARE program is a set of cis-regulatory sequences. CLARE then proceeds to build a classifier. A control set is created by sampling regions from the non-coding portions of the human genome ensuring the balance between the positive and negative datasets in terms of length and GC content so that the classifier does not learn to differentiate between the two sets solely on the basis of GC content. After this, feature vectors are created for all the sequences from both the sets by scanning for matches using the known PWMs as well as 10 over-represented motifs found in the positive set using the PRIORITY [61] tool which employs Gibbs sampling. Once the feature vectors are prepared, they are used to train a linear LASSO machine learning classifier to predict regulatory regions.

The EnhancerFinder [62] tool for the task of identifying tissue specific regulatory regions uses a variety of data that captures details about sequence conservation, DNA sequence in the form of K-mer counts, chromatin accessibility in the form of histone modifications, and TF bindings. It uses different kernel functions for different feature data, and then a weighted combination of these kernel functions is learned using a multi kernel learning (MKL) classifier. EnhancerFinder employs a two step method. At the first step, an MKL classifier is used distinguish enhancers from the background DNA on a genome-wide basis, and at the second step, an MKL classifier is used to map the predicted regions to specific tissues in which they are functional. The authors compared the performance of EnhancerFinder against the performance of CLARE [59] and found it to be significantly better.

Blatti and Sinha *et al.* in their work in [63] approached the problem of identifying cell type specific regulatory regions by using data of gene expression, low resolution chromatin

accessibility, and TF-DNA binding specificities in the form of motif scores. They were able to build reliable models applicable for genome-wide prediction for more than 70 expression domains in the Drosophila embryo. Expression domains can be seen as cell types and different stages of development of the embryo describing the patterns of gene expression.

1.5 Organization of the Thesis

The rest of this thesis is organized as follows:

In Chapter 2, first a bird's eye view of the problem is given. Then the three machine learning methods, namely, hidden Markov models (HMMs), phylogenetic hidden Markov models (phylo-HMMs), and support vector machines (SVMs) are described along with the reasons for using them in this work. To set the premise for discussing phylo-HMMs, multiple sequence alignment, ancestral reconstruction, and phylogenetic models are also discussed.

In Chapter 3, firstly, the data used in this work followed by the data preprocessing steps are defined. Secondly, the problem statement is defined formally in terms of the data. Finally, the specific details of the machine learning models built are described.

In Chapter 4, the results obtained by the different machine learning models are compared and analyzed.

In Chapter 5, we start off by summarizing our work and describe how it can be useful to other researchers. Then we discuss what future extensions to this work can prove beneficial.

CHAPTER 2
General Overview of Methods

The problem statement associated with this thesis will be discussed in detail in the next chapter. However, here before discussing the general overview of the methods we would like to give a bird's eye view of the problem statement. The aim is to build models that can computationally label cell type specific regulatory regions of a given genome using the DNA sequence information of that genome and genomes of related species. This is basically an instance of the problem of sequence labelling or sequence tagging which involves the assignment of a categorical label to each member of a sequence of observed values [64].

2.1 Hidden Markov Models

The machine learning algorithms dealing with sequence labelling tasks usually do not assign labels to observations by treating them as independent quantities but also take into consideration the effect of the nearby observations. Many of the machine learning models generally used for sequence labelling are probabilistic in nature. Hidden Markov models (HMMs) are popular probabilistic sequence labelling classifiers. They are used in situations where the data is a sequence of units or observations depending probabilistically on the state of a stochastic system or process. A general way of saying this is that the observations are emitted from a state. The true state of the system is not available and is the latent or hidden causal variable that needs to be inferred. The way an HMM works is that at any index $i = 0, 1, 2, ...$ the system is in some latent state $S_i = s$ and it stochastically emits

an observation $O_i = o$ based only on s. The system then probabilistically transitions to a new state S_{i+1} according to a probability distribution $\mathsf{P}[S_{i+1}|S_i]$ and the process continues in this manner. The first order HMM which is used in this work is based on two simplifying assumptions. First, as shown in equation 2.1 the probability of transitioning to a state at the next index only depends on the state at the current index and not on the states visited by the system before. This is called as the Markovian assumption.

$$\mathsf{P}[S_{i+1}|S_i, S_{i-1}, S_{i-2}, ...] = \mathsf{P}[S_{i+1}|S_i] \tag{2.1}$$

Second, as depicted in equation 2.2 the observation O_i emitted by state S_i only depends on the state S_i and is independent of all other states and observations. Later, we will see how this condition has been relaxed to a certain extent to better suit the problem at hand.

$$\mathsf{P}[O_i|S_0, ..., S_i, S_{i+1}, ..., O_0, ..., O_i, O_{i+1}, ...] = \mathsf{P}[O_i|S_i] \tag{2.2}$$

The index $i = 0, 1, 2, ..$ mentioned earlier can represent a time slice if the process is running through time or can be a position if the process is running through space, for example, along the length of a DNA sequence emitting nucleotides as observations.

The finite first order variant of HMM used in this work can be defined as a 5-tuple model which consists of:

- $S-$ finite set of states
- $O-$ finite set of observations
- State Transition Probabilities $-P_{s,s'} = \mathsf{P}[S_{i+1} = s'|S_i = s], \quad \forall s', s \in S$. This is represented concisely in the form of a transition matrix T of size $|S| \times |S|$, where the

j^{th} row gives the multinomial distribution for the next state over all the states in S given the current state is j.

- Observation Emission Probabilities $-b_{s,o} = \mathsf{P}[O_i = o|S_i = s], \forall o \in O$ and $s \in S$ which are stored in an emission matrix E of size $|S|\times|O|$, where the j^{th} row gives the multinomial distribution over all the observations in O given the system is in state j.
- $\Pi-$ Initial state probability vector, where Π_j is the probability for the HMM to start at state j

The model of the HMM just described above can be used to generate a sequence of observations$(O_0, O_1, O_2, ..., O_{L-1})$ of length L in the following way:

1. Set $i = 0$ and pick an initial state S_0 as per the initial state probability vector Π.
2. As per the emission probability distribution for the state S_i emit an observation O_i.
3. Transit to a state S_{i+1} chosen randomly from the transition probability distribution given by the current state S_i.
4. Set $i = i + 1$. If $i < L$ then goto step 2, else exit.

Keeping this vision of the HMM as a generative model three basic problems that make HMMs relevant to real world applications can be solved. These problems are discussed in the following subsection.

2.1.1 Inference and Learning Problems

Decoding: Finding the most likely state sequence

The decoding problem puts forward the objective to find the sequence of states that best explains a given observation sequence $O_0, O_1, O_2, ..., O_{L-1}$.

$$\arg \max_{S_0,...,S_{L-1}} \mathsf{P}[S_0, S_1, ..., S_{L-1}|O_0, O_1, ..., O_{L-1}] \tag{2.3}$$

The Viterbi algorithm [65] is used to solve the decoding problem. The state sequence returned by the Viterbi algorithm is called as the Viterbi path. The running time of the Viterbi algorithm is $O(|S|^2 \cdot L)$

Posterior Decoding

In posterior decoding the aim is to find the posterior probability of the process being in state s at some i^{th} index for the sequence $O_0, O_1, O_2, ..., O_{L-1}$, given mathematically as $\mathsf{P}[S_i = s|O_0, O_1, ..., O_{L-1}]$. This helps us in finding the posterior distribution at index i over all the states, and the maximum a posteriori probability and hence the most likely state can be found out by simply taking the maximum over all these posterior probabilities.

The posterior decoding problem is addressed using the forward-backward algorithm [66]. The output of the forward-backward algorithm consists of a vector of size $|S|$ for every position of the input observation sequence representing the posterior predictive distribution over all the states at that position. The output of the forward-backward algorithm is often more useful than the Viterbi algorithm as it gives a quantitative measure of confidence in the predictions made, as opposed to a single label, allowing one to filter the results as per the required confidence threshold. The running time of the forward-backward algorithm is $O(|S|^2 \cdot L)$

Learning of Model Parameters

There are two main approaches for learning the model parameters for an HMM from given data: the Baum-Welch algorithm [67] and maximum likelihood learning from complete trajectories. The Baum-Welch algorithm is effective for learning the HMM parameters when data consists of observation sequences and the information regarding the

state paths for these sequences is missing. The Baum-Welch algorithm is an expectation-maximization (EM) algorithm, and is described below.

Given an observation sequence $O_0, O_1, ..., O_{L-1}$, the Baum-Welch algorithm learns the model parameters of the HMM, namely the state transition probabilities, the emission observation probabilities, and the initial state probabilities. It does so by running an EM procedure that aims at maximizing $\mathsf{P}[O_0, O_1, ..., O_{L-1}]$ w.r.t. the model parameters. It starts with an initial guess for the model parameters and then repeats the following EM procedure until convergence:

- E-step:
  1. Compute $\mathsf{P}[S_i = s|O_0, O_1, ..., O_{L-1}], \forall s \in S$ and $0 \leq i < L$ using the forward-backward algorithm [66].
  2. Compute $\mathsf{P}[S_i = s, S_{i+1} = s^{'}|O_0, O_1, ..., O_{L-1}], \forall s, s^{'} \in S$ and $0 \leq i < L - 1$.
- M-step:
  1. Compute $\Pi_s = \mathsf{P}[S_0 = s|O_0, O_1, ..., O_{L-1}], \forall s \in S$ using the forward-backward algorithm [66].
  2. Compute state transition probabilities $\forall s, s^{'} \in S$ as follows:

$$\begin{aligned} P_{s,s'} &= \frac{\text{Expected \# } s \text{ to } s^{'} \text{transitions}}{\text{Expected \# transitions from } s} \\ &= \frac{\sum_{i=0}^{L-2} \mathsf{P}[S_i = s, S_{i+1} = s^{'}|O_0, O_1, ..., O_{L-1}]}{\sum_{i=0}^{L-2} \mathsf{P}[S_i = s|O_0, O_1, ..., O_{L-1}]} \end{aligned} \tag{2.4}$$

3. Compute observation emission probabilities as follows:

$$b_{s,o} = \frac{\text{Expected \# } o \text{ emitted from } s}{\text{Expected \# occurrences of } s}$$
$$= \frac{\sum_{i=0}^{L-1} \mathbb{1}\{O_i = o\} \cdot \mathsf{P}[S_i = s | O_0, O_1, ..., O_{L-1}]}{\sum_{i=0}^{L-1} \mathsf{P}[S_i | O_0, O_1, ..., O_{L-1}]} \tag{2.5}$$

Note that here we have described the Baum-Welch algorithm for the case when we have only one observation sequence but the algorithm can be easily extended to accommodate for the case when multiple observation sequences are available. Like all EM algorithms, the Baum-Welch algorithm is vulnerable to getting stuck in local optima and hence running the Baum-Welch algorithm multiple times with a different set of initial parameters, and then picking the best solution among the results of these runs is a common practice. The time complexity for one EM iteration of the Baum-Welch algorithm is $O(|S|^2 \cdot L)$.

When we have data in the form of trajectories that contain observation sequences along with their true state paths then instead of running an EM algorithm, we can learn model parameters that maximize the likelihood of the given data by using the procedure defined below.

Given a set ξ of n sample trajectories, with the k^{th} trajectory of the form $\xi_k = (S^k_0, O^k_0), (S^k_1, O^k_1), ..., (S^k_{L_k-1}, O^k_{L_k-1})$ that includes both observation and true system state, the maximum likelihood estimation of the parameters of the HMM can be calculated as shown through equations 2.6 to 2.8. For simplicity of notation $\xi_k(S_i)$ and $\xi_k(O_i)$ are used to refer to the state and observation at the i^{th} state of the k^{th} trajectory.

$$\Pi_s = \frac{\text{\# trajectories starting in } s}{n} = \frac{\sum_{k=0}^{n-1} \mathbb{1}\{\xi_k(S_0) = s\}}{n} \tag{2.6}$$

$$P_{s,s'} = \frac{\text{\# } s \text{ to } s' \text{ transitions}}{\text{\# transitions from } s} = \frac{\sum_{k=0}^{n-1} \sum_{i=0}^{L_k-2} \mathbb{1}\{\xi_k(S_i) = s, \xi_k(S_{i+1}) = s'\}}{\sum_{k=0}^{n-1} \sum_{i=0}^{L_k-2} \mathbb{1}\{\xi_k(S_i) = s\}} \tag{2.7}$$

$$b_{s,o} = \frac{\text{\# } o \text{ emitted from state } s}{\text{\# occurrences of state } s} = \frac{\sum_{k=0}^{n-1} \sum_{i=0}^{L_k-1} \mathbb{1}\{\xi_k(S_i) = s, \xi_k(O_i) = o\}}{\sum_{k=0}^{n-1} \sum_{i=0}^{L_k-1} \mathbb{1}\{\xi_k(S_i) = s\}} \tag{2.8}$$

2.1.2 Approach

A DNA sequence having different components like regulatory regions and genes interspersed in background DNA can be modeled as being generated by an HMM working along the length of the sequence, transiting between states associated with the different components and emitting observations in the form of nucleotides or groups of nucleotides. The parameters of such an HMM can be learned, given a set of sequences along with their true internal state parse as mentioned above.

2.2 Phylogenetic Tree

As mentioned in section 1.1.7 all organisms have descended from a single common ancestor and among any set of species there exists an evolutionary relationship which in the literature is termed as *phylogeny* [68]. A *phylogenetic tree* as shown in Figure 2–1 is used to represent phylogeny among a set of species. The leaves of a phylogenetic tree represent the extant species while the internal nodes represent the hypothetical evolutionary ancestors of the extant species. The branching pattern of the tree represents how the

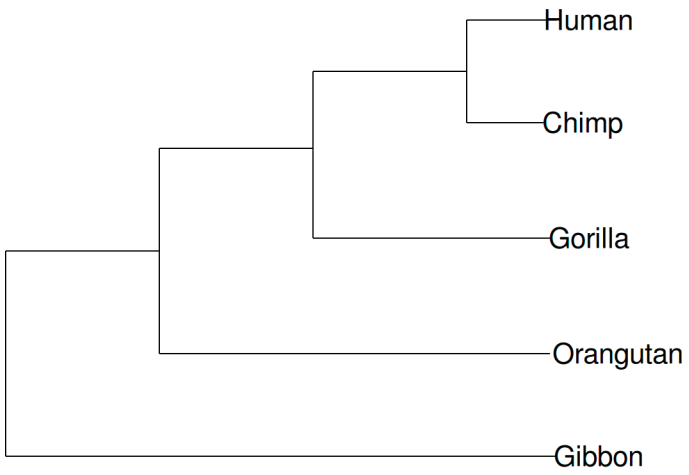

Figure 2–1: Phylogenetic Tree
The figure is of a subtree of the tree containing 100 leaf nodes that was used for building the ”Vertebrate Multiz Alignment & Conservation (100 Species)” track of the UCSC Genome browser (https://genome.ucsc.edu/) [69, 70].

species have evolved from a sequence of common ancestors. Each branching point represents a speciation event, and the branches represent the evolutionary derivations. The length of a branch represents the amount of mutations accumulated between the species. Two extant species on the tree are said to be more closely related if they have a more recent common ancestor and are said to be less closely related otherwise.

2.3 Multiple Sequence Alignment

Multiple sequence alignment (MSA) is an important step in representing the similarity between DNA sequences. The conserved bases are aligned in the same column of the alignment by inserting gaps (”−”) within the sequences to accommodate for insertions and deletions. One of the most common approaches to MSA is that of progressive alignment. Progressive alignment is a heuristic approach based on a simple idea. Firstly, align pairs of most closely related sequences and then align the alignments to get alignments for a larger

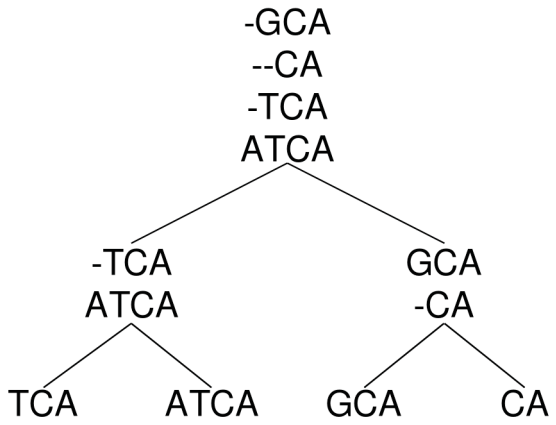

Figure 2–2: Progressive Multiple Sequence Alignment

number of sequences. The order in which the pairwise alignments are to be performed is decided by a guide tree [71]. The pairwise alignment starts at the leaves and then proceeds towards the root. At each level, the sequences or alignments found at the siblings are aligned as shown in Figure 2–2. A phylogenetic tree connecting multiple species can act as a guide tree to progressively align the genomes of species represented by the leaves. The alignment is decided upon so as to maximize the similarity score among the given sequences under a given scoring scheme. For example, the scoring scheme used for the alignment shown in Figure 2–2 is +1 for a match, -1 for a mismatch, and a gap penalty of -2. Progressive MSA is not guaranteed to achieve the global optimal score as decisions made in the earlier stages without the knowledge of all the sequences can lead to errors that are propagated all the way through to the final outcome. However, it is widely used because it is fast and can easily scale to thousands of sequences [71].

There are a large number of progressive MSA programs such as CLUSTAL W [72], T-Coffee [73], MUSCLE [74], and MULTIZ [75]. Here, we will stick to discussing MULTIZ

as the alignments used in this work which will be introduced later are generated using MULTIZ.

The MULTIZ program is capable of generating genome-wide MSA given the DNA sequences of a set of species and the phylogenetic tree that connects them. The pairwise alignments are computed using the BLASTZ program [76, 77] or its improvement, the lastz program [78]. These are local alignment programs. A local alignment of two DNA sequences is different from a global alignment such that in a local alignment one of the sequences known as the query sequence is aligned to the best matching substring of the other sequence known as the reference sequence. The BLASTZ program follows three main steps:

1. It looks for short nearly exact matches in the two sequences.
2. Each of the alignments found in step 1 is extended in either direction without allowing for gaps. This extension stops once the alignment score starts to fall below a certain threshold. Provided the gap-free alignments scored beyond a decided threshold, they are further extended, this time allowing for gaps. Again, the alignments assigned scores beyond a particular value are retained while the others are discarded.
3. To have more connectivity in the alignments, steps 1 and 2 are repeated for the intervening regions between the alignments obtained after step 2, but with somewhat lower requirements.

At the end of the three step procedure, there might be cases where multiple regions may align to the same region. This can be dealt with techniques such as chaining, in which only the matches that follow same relative ordering in both the sequences are retained, and techniques that only retain the best scoring alignments with the constraint that every

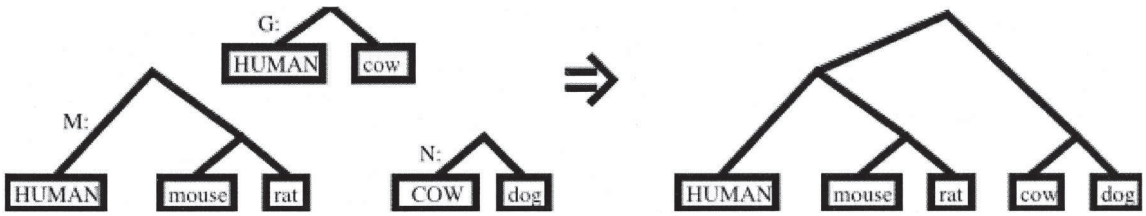

Figure 2–3: Working of MULTIZ
This figure is taken from [75]. Here, M and N are two blocksets referenced by human and cow respectively. G is the blockset corresponding to the human referenced pairwise alignment of human and cow that guides the alignment procedure to get the human referenced blockset for the set of species given by the union of species present in the blocksets M and N.

position in the reference sequence can appear in at most one alignment [77]. The MULTIZ program gets a set of local alignments from the BLASTZ program in the form of a set of blocks referred to as a blockset by the authors of MULTIZ. Every blockset has a reference sequence. In order to align two blocksets, for instance, one referenced by human and the other by cow to get a blockset referenced by human, MULTIZ uses a blockset of pairwise alignment between human and cow referenced by human to guide the procedure of alignment which uses a dynamic programming technique [75]. This is pictorially represented in Figure 2–3 that is taken from [75].

2.4 Ancestral DNA Sequence Reconstruction

Ancestral sequence reconstruction attempts to predict the DNA sequences of the ancestral species using a molecular evolution model fed with the DNA sequence information of their modern day descendants [79]. The primary motivation to address the problem of ancestral reconstruction was to develop a better understanding of the evolutionary patterns. The knowledge about evolutionary patterns plays an integral role in addressing many problems such as prediction of protein coding regions, and prediction of regulatory

regions [80]. Given a set of orthologous genomic DNA sequences along with the phylogenetic tree relating them, the identification of the DNA sequences of all the ancestral species in the tree involves the following major steps as mentioned by Blanchette *et al.* in [81]:

1. Multiple Sequence Alignment (MSA) of the given set of orthologous DNA sequences, which helps in identifying the regions derived from a common ancestral sequence either through direct inheritance or substitution.
2. Indel reconstruction. Indel is the term used to refer to both insertion and deletion of bases in a DNA sequence. Indel reconstruction can be defined as inferring the history of insertions and deletions that occurred along the branches of the given phylogenetic tree. The sets of insertions and deletions are selected such that gaps present in the MSA are best explained under the assumed evolutionary parameters.
3. Substitution reconstruction, which involves predicting which nucleotide was present at each position in the ancestral sequence.

Ancestors 1.0 [79] is a program provided as a web server for ancestral reconstruction that performs all of the above steps. It brings the tools and algorithms developed by different authors under one umbrella. It takes as input a set of DNA sequences in Fasta format [82] along with the phylogenetic tree relating them, in Newick format [83], and outputs an alignment containing both, the provided DNA sequences of extant species as well as the predicted ancestral DNA sequences. Along with the extended alignment it also outputs the most likely indel reconstruction and the posterior probability for each position of each ancestor. One can also provide aligned sequences and choose whether he/she wants them to be realigned. Ancestors 1.0 provides different MSA programs to choose from. These

include MUSCLE [74] and TBA [75]. For indel reconstruction it runs the algorithms developed by Diallo *et al.* [84] to solve the indel maximum likelihood problem, and for inferring substitution it uses an adaptation of the Felsenstein algorithm [85, 86].

2.5 Phylogenetic Models

HMMs, discussed in the previous section 2.1 can exploit a given DNA sequence to uncover regulatory regions using the different inference algorithms. However, it cannot directly make use of sequence evolution. As discussed earlier in section 1.1.7, knowledge about sequence conservation can play an important role in identifying the regulatory regions. One simple and obvious extension of an HMM to take into consideration sequence evolution might be to change the set of observations O to have alignment columns, but this technique does not scale beyond a few species as the number of possible observations increases exponentially with the number of species. On the other hand, the emission probability of an alignment column can be calculated in an elegant manner using phylogenetic models. Phylogenetic models are probabilistic models developed for modeling the process of sequence evolution as a Markov process. In order to calculate the likelihood of an alignment column, phylogenetic models along with the topology and branch lengths of the phylogenetic tree connecting the present day species, also take into consideration the patterns of substitution and the background distribution of the characters. Each character in an alignment column belongs to the alphabet set Σ (for example, $\Sigma = \{A, C, G, T, " - "\}$) with the constraint that every character in a column should not be "−". Formally, a phylogenetic model can be defined as $\psi = (Q, \tau, \beta, \pi)$ where:

- $Q-$ substitution rate matrix of size $|\Sigma| \times |\Sigma|$.

- $\tau-$ tree topology. The tree, is a binary tree with n leaves, $n-1$ internal nodes, and $2n-2$ edges called branches.
- $\beta-$ vector of branch lengths for the tree. A branch length is a non-negative real number that signifies the expected number of substituions per site.
- $\pi-$ vector of size $|\Sigma|$ containing background frequencies.

A phylogenetic model ψ can be seen as modeling the process of evolution as a Markov process and generating an alignment column in the following manner:

1. A character is picked randomly from Σ according to the background distribution π and is placed at the root of the tree.
2. As the process moves along the branches of the tree from the root to the leaves, characters are assigned to the nodes depending on the substitution rates given by the substitution rate matrix Q, the length of the branch between the node and its parent, and the base present at the parent node. The probability of substituting a character a by character b over a branch of length l denoted as $\mathsf{P}[b|a,l]$ can be obtained by looking up the element given by the row corresponding to character a and the column corresponding to character b in the substitution probability matrix for branch length l given by equation 2.9:

$$\mathsf{P}[l] = e^{Ql} \tag{2.9}$$

where, $e^{Ql} = \sum_{i=0}^{\infty} \frac{(Ql)^i}{i!}$ [11]

In this manner, the substitution process is modeled as a continuous-time Markov process, continuous time because the branch lengths represent evolution time which is continuous and Markov because the probability of a character appearing at a node given the

character at its parent node is independent of the characters present at the parent's ancestors.

As described in [87], the likelihood of an alignment column X_i present at some arbitrary position i of an alignment X of DNA sequences from species represented at the leaves of a phylogenetic tree τ can be calculated under a phylogenetic model $\psi = (Q, \tau, \beta, \pi)$ by marginalizing over all the possible labelings of the ancestral nodes present in the tree. Thus, the likelihood of column X_i is $\mathsf{P}[X_i|\psi] = \sum_\zeta \mathsf{P}[\zeta, X_i|\psi]$, where ζ is the labeling of all the ancestral nodes. The number of possible labelings grows exponentially in the number of leaf nodes. For n leaves, the number of ancestral nodes in the tree is $n - 1$ and there are $|\Sigma|^{n-1}$ possible labellings. This makes the computation time of the likelihood exponential in the number of leaves for a constant $|\Sigma|$. Felsenstein [85, 86] provided an algorithm that uses dynamic programming to efficiently perform the marginalizing step as follows. For a node u with children v and w connected via edges of length l_v and l_w respectively, the probability of all the leaves present below u given the character at node u is a, is denoted by $\mathsf{P}[L_u|a]$ which can be evaluated using equation 2.10.

$$\mathsf{P}[L_u|a] = \begin{cases} \sum_b \mathsf{P}[b|a, l_u]\mathsf{P}[L_v|b] \sum_c \mathsf{P}[c|a, l_w]\mathsf{P}[L_w|c], & \text{if } u \text{ is a non-leaf node} \\ \mathbb{1}\{x_u = a\}, & \text{if } u \text{ is a leaf node} \end{cases} \tag{2.10}$$

The total probability of any alignment column X_i is given as $\mathsf{P}[X_i|\psi] = \sum_a \pi_a \mathsf{P}[L_r|a]$, where π_a denotes the background probability of character a and r denotes the root of the tree. The running time of the Felsenstein algorithm is $O(n \cdot |\Sigma|^2)$. In case if X is an alignment of extant as well as ancestral sequences, as presented by the output of the Ancestors 1.0 [79]

program, then in order to calculate the likelihood of any alignment column X_i, there is no need for marginalizing over the ancestral labellings and the likelihood can be computed in $O(n)$ time using equation 2.11 given below:

$$\mathsf{P}[X_i|\psi] = \mathsf{P}[x_r|\psi] \prod_{(u,v)\in E_\tau} \mathsf{P}[x_v|x_u, \psi] \tag{2.11}$$

where,

- x_u is the notation used to represent the character present at node u, and r is the notation used for the root node.
- E_τ is the set of edges of tree τ. Each edge is in the form of a tuple given as (parent node, child node). In a tree with n leaves there are $2n - 2$ edges.

2.6 Phylogenetic Hidden Markov Models

Phylogenetic hidden Markov models (phylo-HMM) [87] are a variant of the HMM specially designed to exploit the knowledge of sequence evolution along with DNA sequence information. In phylo-HMMs both phylogenetic models and HMMs are combined. Thus, phylo-HMMs are a combination of two Markov processes, one running along the branches of a phylogenetic tree and the other running along the length of a DNA sequence. As described by Yang [88], HMMs operating on DNA sequences work in the dimension of space; whereas, phylogenetic models work in the dimension of time.

2.6.1 Mathematical Model of Phylogenetic Hidden Markov Models

A phylo-HMM is formally defined as a 5-tuple model consisting of the following:

- $S-$ finite set of states
- $\Sigma-$ alphabet set. Every member of an alignment column belongs to the alphabet set.

- $\Psi = \{\psi_0, ..., \psi_{|S|} - 1\}-$ set of phylogenetic models associated with the states, where ψ_s is associated with state s. Different phylogenetic models are used with different states as the substituion rates, patterns of substitution, and background distributions may differ from state to state, for instance, states corresponding to functional and non-functional DNA.
- State Transition Probabilities $-P_{s,s'} = \mathsf{P}\{S_{i+1} = s'|S_i = s\}, \quad \forall s', s \in S$. These are represented concisely in the form of a transition matrix T of size $|S| \times |S|$.
- $\Pi-$ initial state probability vector, where Π_j is the probability for the HMM to start at state j.

2.6.2 Inference and Learning Problems

Inference Problems

Given an alignment X of length L, which can be conveniently viewed as a sequence of alignment columns $X_0, X_1, ..., X_{L-1}$, the decoding problem and the posterior decoding problem described in section 2.1.1 can be solved using the Viterbi [65] and the forward-backward [66] algorithms respectively, as in case of a regular HMM. Alignment columns are analogous to observations being emitted from the states in case of a regular HMM and the emission probability of an alignment column can be seen as its likelihood under the phylogenetic model associated with the state. The running time of the Viterbi and the forward-backward algorithms for a tree with n leaves increases to $O(|S|^2 \cdot L \cdot n \cdot |\Sigma|^2)$ or $O(|S|^2 \cdot L \cdot n)$ depending on whether the ancestral sequences are available or not.

Learning of Model Parameters

The model parameters of a phylo-HMM can be learned using the maximum likelihood estimation technique, given a phylogenetic tree τ with m nodes, and a set ξ of n

sample trajectories, where each trajectory consists of an alignment of DNA sequences from m species and the state parse corresponding to the alignment. Trajectory ξ_k is of the form $\xi_k = (X^k, S^k)\ \forall 0 \leq k \leq n-1$. Further X^k and S^k can be seen as a vector of alignment columns, $X_0^k, X_1^k, ..., X_{L_k-1}^k$, and vector a state labels, $S_0^k, S_1^k, ..., S_{L_k-1}^k$, where L_k is the number of columns in the k^{th} alignment. The notation $X_{i,j}^k$ is used to denote the character present at the row corresponding to the j^{th} node in τ and the i^{th} column of alignment X^k. The initial state probabilities and the state transition probabilities can be learned as in case of the regular HMM; however, for completeness the expressions are mentioned below:

$$\Pi_s = \frac{\text{\# alignments starting in state } s}{n} = \frac{\sum_{k=0}^{n-1} S_0^k = s}{n} \tag{2.12}$$

$$P_{s,s'} = \frac{\text{\# } s \text{ to } s' \text{ transitions}}{\text{\# transitions from } s} = \frac{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-2} \mathbb{1}\{S_i^k = s, S_{i+1}^k = s'\}}{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-2} \mathbb{1}\{S_i^k = s\}} \tag{2.13}$$

The likelihood of a column X_i under the phylogenetic model ψ_s given by $\mathsf{P}[X_i|\psi_s]$ can also be written as $\mathsf{P}[X_i|s]$ because ψ_s is followed when the process is in state s. Therefore, equation 2.11 can be rephrased as:

$$\mathsf{P}[X_i|\psi_s] = \mathsf{P}[X_i|s] = \mathsf{P}[x_r|s] \prod_{(u,v)\in E_\tau} \mathsf{P}[x_v|x_u, s] \tag{2.14}$$

The probabilities $\mathsf{P}[x_r = a|s]$ and $\mathsf{P}[x_v = b|x_u = a, \psi_s]$ can be estimated as follows:

$$\mathsf{P}[x_r = a|s] = \frac{\text{\# columns emitted from s with } a \text{ at root}}{\text{\# alignment columns emitted from } s} = \frac{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-1} \mathbb{1}\{X_{i,r}^k = a, S_i^k = s\}}{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-1} \mathbb{1}\{S_i^k = s\}} \tag{2.15}$$

$$\mathsf{P}[x_v = b|x_u = a, s] = \frac{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-1} \mathbb{1}\{X_{i,u}^k = a, X_{i,v}^k = b, S_i = s\}}{\sum_{k=0}^{n-1}\sum_{i=0}^{L_k-1} \mathbb{1}\{X_{i,u}^k = a, S_i = s\}} \tag{2.16}$$

Note that the Baum-Welch algorithm would have to be used in scenarios where information about the true state paths is absent.

2.7 Support Vector Machine

Support Vector Machine (SVM) is a popular supervised machine learning model that can be used to solve classification as well as regression problems. Since here we are concerned with solving a classification problem, we will stick to discussing SVM as a classifier. SVM is a decision machine and unlike probabilistic classifiers does not give posterior probabilities as output [89]. Considering the two-class case, given a set of training data points along with their true labels, an SVM learns a decision boundary in the form of a hyperplane. The hyperplane that is chosen should not only fit the training data but should also have low generalization error. To tackle this problem SVM chooses a hyperplane that maximizes the margin. The margin for a given hyperplane is defined as twice the Euclidean distance between the hyperplane and the nearest training data point as shown in Figure 2–4. The running time to train an SVM is quadratic in the number of examples, and

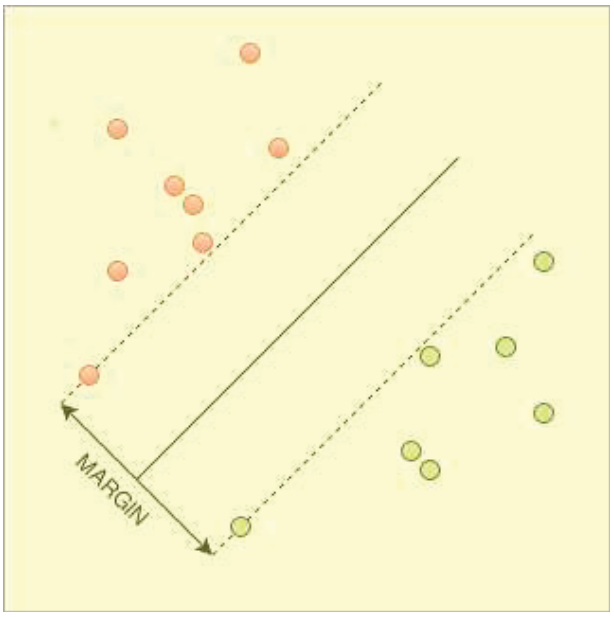

Figure 2–4: SVM Margin

hence, it is computationally expensive to train SVMs when the size of the training dataset exceeds beyond a few hundred thousand examples.

The SVM is able to only choose linear decision boundaries. However, it can separate non-linear data by choosing linear decision boundaries in some higher dimensional space which will correspond to non-linear decision boundaries in the input space. This can be achieved with the help of kernel functions [89]. Kernel functions are used for computing similarity between higher dimensional mappings of two data points without actually mapping them.

There are mainly two flavours of SVM: the hard-margin SVM and the soft-margin SVM.

Hard-margin SVM tries to learn a decision boundary which maximizes the margin with the constraint that no misclassifications should be encountered when the model is applied to the training data.

Soft-margin SVM, unlike the hard-margin SVM, does allow for some training data points to be misclassified in order to prevent overfitting of data. The hard-margin SVM may result in highly curvy decision boundaries with narrow margins in the attempt of classifying every training example correctly. This may lead to high generalization error from overfitting on training data. To check on the misclassifications, the soft-margin SVM penalizes the misclassified points. The penalty ζ_i associated with a misclassified training example i grows linearly with the distance of the point from the decision boundary. The sum of all such penalties is termed as *soft-error*. The soft-margin SVM solves an optimization problem that combines both the maximization of the margin and the minimization of the soft-error. The degree to which the misclassifications are penalized is controlled by a parameter usually denoted by C. The higher C is, the more are the misclassifications penalized. Consequently, C acts as a knob for controlling overfitting and is usually selected via cross-validation strategies.

One important difference between SVM and the HMM is that unlike the HMM, SVM assumes the data to be independent and identically distributed (i.i.d). This i.i.d assumption is not true in case of sequential data where the order of the observations matter. Hence, some tricks had to be adapted to use SVM for the sequence labelling task. These tricks as well as the reason behind choosing SVM for this particular problem will be discussed in the subsequent chapters.

CHAPTER 3
Method Development and Application

In this chapter we will describe the data used for the methods discussed in the previous chapter and also discuss the implementation details of those methods.

3.1 Data

This section highlights the different pieces of data used by the methods described in the previous chapter. The description of the data along with its sources is given below.

- The latest build of the human reference genome, the GRCh38/hg38 (Dec. 2009) assembly produced by the Genome Reference Consortium [90]. It was downloaded from the UCSC Genome Browser (http://genome.ucsc.edu/) [69, 70]. It is divided into a total of 24 chromosomes, the chromosomes numbered through 1 to 22, and the chromosomes X and Y. The total length of the human reference genome is close to three billion bases.
- Set of regulatory regions represented as genome coordinates in BED format which is discussed in [91]. This set of regions was presented by the ENCODE Analysis Working Group. It has been derived from the results of a large number of ChIP-seq experiments performed by the ENCODE project [92], covering a broad spectrum of 161 TFs and 91 human cell types [93], and was downloaded from the ENCODE data repository at the UCSC Genome Browser [93, 94]. It can be viewed as the ”Txn Factor ChIP Track” in the UCSC Genome Browser. The genome coordinates present in this piece of data are as per the GRCh37/hg19 (Feb. 2009) assembly of the human

genome and had to be converted to match the corresponding coordinates in the hg38 assembly. This was achieved by using the liftOver tool from the UCSC Kent tools [95]. Following the lift over, the set of regions was divided into two subsets. One containing regulatory regions functional in the GM12878 lymphoblastoid cell line [96], which will be referred to as GM12878 regulatory regions in the rest of the thesis, and the other containing the set of regions that are functional in one or more cell types other than GM12878 lymphoblastoid cell line, which for simplicity will be referred to as non-GM12878 regulatory regions from here onwards. The GM12878 regulatory regions cover $\sim 1.7\%$ of the total genome, while the non-GM12878 regulatory regions cover $\sim 10\%$.

- Set of positions of coding exons downloaded from the "Old UCSC Genes" track of the UCSC Genome Browser (http://genome.ucsc.edu/) [69, 70]. The coding exons cover $\sim 1\%$ of the genome.
- Genome wide MSA of 115 species, 58 present day mammals and 57 ancestral species. This was achieved by feeding the Ancestors 1.0 [79] program with the Hg38 100-way alignment downloaded from the UCSC Genome Browser [69, 97]. The Hg38 100-way alignment is an alignment of 100 vertebrate species [98] aligned using the MULTIZ program [75] keeping human as the reference species. The output of the Ancestors 1.0 was then filtered to keep only the mammals and their ancestors. The rationale behind this filtering step was that sequence conservation may loose its discriminatory power to identify regulatory regions if the species considered are at large evolutionary distances. This is because even regulatory regions

may accumulate large number of mutations over large evolutionary distances. We will refer to this piece of data as the 115 mammal alignment.

3.2 Problem Definition

Now that a detailed description of the data being used is provided, the premise for discussing the problem statement in detail is set. The problem statement is to build models that can computationally identify cell type specific regulatory regions within the genome of some concerned species, given the DNA sequence information of both the concerned and its related species. In this work we train the models to identify regulatory regions in the GM12878 lymphoblastoid cell line in humans. The GM12878 regulatory regions form the positive class that we want to learn to identify. The set of coding exons acts as a positive control as in the work of Elnitski *et al.* [54]. The negative set comprises all the DNA excluding the GM12878 regulatory regions and including the coding exons as well as the non-GM12878 regulatory regions; which we will refer to as the non-functional DNA.

3.3 Data Preprocessing

DNA sequences of all chromosomes of the hg38 human genome assembly, originally represented as a sequence of nucleotides were represented as a sequence of 6-mers using a sliding window of size six which was slid by a step-size of one. The coordinates consisting of the start and end positions of the GM12878 regulatory regions, non-GM12878 regulatory regions, and set of coding exons were modified to adapt to the 6-mer representation of the chromosomes. The start coordinate of a region was replaced by the starting position of the first window containing the start coordinate of the region, and the end coordinate was replaced by the starting position of the last window containing the end coordinate. The

first window of which the start coordinate denoted by p_{start} is a part of, begins at position $\max(p_{start} - 5, 0)$, and the last window containing the end coordinate denoted as p_{end} is the window beginning at position $\min(p_{end}, \text{last position} - 5)$, where the last position is one less than the size of the chromosome under consideration. All regions belonging to the non-GM12878 regulatory or the coding exons set that had an overlap of at least one base with a region in the GM12878 regulatory set were removed. This was done to give preference to regions belonging to the GM12878 regulatory set as the objective of this work is to build a classifier which is able to predict GM12878 regulatory regions. The preprocessing of the 115 mammal alignment was done in the following manner:

1. All the alignment columns having a gap in the human sequence were removed.
2. A sliding window of size six was used to represent the alignment as an alignment of sequences of 6-mers. Every 6-mer that contained one or more gaps " – " was counted as a special " – " k-mer. This reduces the set of possible characters to $\{A, C, G, T\}^6 \cup \{\text{" – " k-mer}\}$.

The rationale behind representing the human and the 115 mammal alignment as sequences or alignments of 6-mers was borrowed from [57] that even though we do not have TF binding motifs to discriminate between regulatory and non-regulatory DNA, our classifiers would identify discriminating 6-mers that cover portions of TF binding motifs and learn to identify regulatory regions based on them.

3.4 Learning

The preprocessed data was used to train the different models, which are described here.

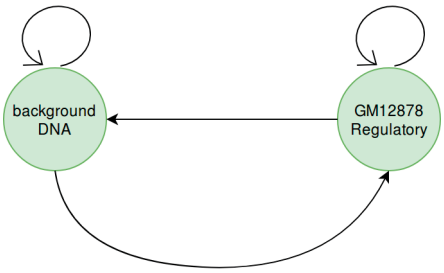

Figure 3–1: Two State HMM

3.4.1 Hidden Markov Models

Regular as well as phylogenetic HMMs with three different structures were trained. The HMMs with different structures use the different pieces of data differently.

Two-State HMM

The two-state HMM as shown in Figure 3–1 has two states: the GM12878 regulatory state and the background DNA state. It is based on an assumption that the GM12878 regulatory regions are generated by the GM12878 regulatory state and the rest of the DNA is generated by the background DNA state. In other words the process running along the genome (or alignments of genomes) is in the GM12878 regulatory state for the GM12878 regulatory regions and is in the background DNA state for the all the other portions of the genome irrespective of whether they fall under the set of coding exons, non-GM12878 regulatory regions or non-functional DNA. Thus, in case of a two-state HMM the 6-mer sequences (alignments in case of phylogenetic HMM) falling within the boundaries of the GM12878 regulatory regions were used to learn the emission probabilities of the GM12878 regulatory state and those falling outside the GM12878 regulatory regions were used to learn the emission probabilities for the background DNA state.

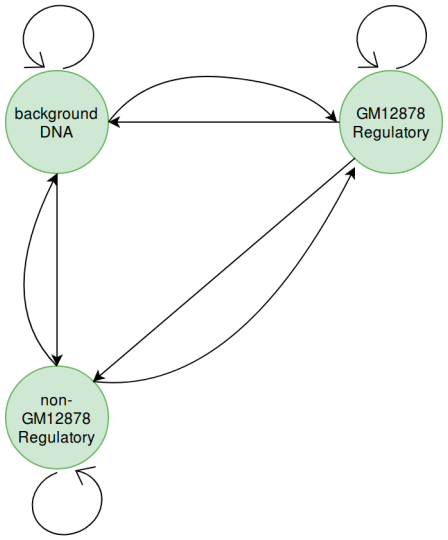

Figure 3–2: Three State HMM

Three-State HMM

The state space associated with the three-state regular and phylogenetic HMM as seen from Figure 3–2 comprises of the GM12878 regulatory state, the non-GM12878 regulatory state, and the background DNA state. The GM12878 regulatory state and the non-GM12878 regulatory state model the GM12878 regulatory and non-GM12878 regulatory regions respectively, whereas the coding exons and the non-functional DNA are assumed to be generated while the process is in the background DNA state.

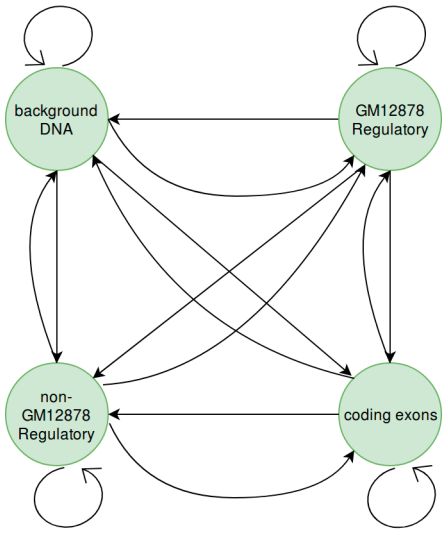

Figure 3–3: Four State HMM

Four-State HMM

As shown in Figure 3–3, the four-state HMM uses four different states to model the four kinds of regions we have, namely, the GM12878 regulatory regions, the non-GM12878 regulatory regions, the coding exons, and the non-functional DNA. The background DNA state, in this case, models the non-functional DNA regions only. It is important to note that the set of regions modelled by the background DNA state changes depending on the structure of the HMM used.

The HMMs mentioned above are modelled as generators of DNA sequences in case of regular HMMs and as generators of alignments in case of phylo-HMMs. The regular HMM generates DNA sequences by emitting overlapping 6-mers as observations; whereas on the other hand, the phylogenetic HMM generates alignments by emitting alignment columns of 6-mers. Figure 3–4 shows the observations in the form of 6-mers that are emitted by a regular HMM to generate the given DNA sequence. As the HMMs in our

Position i	Observations Emitted							
0	A	A	T	T	G	G		
1		A	T	T	G	G	C	
2			T	T	G	G	C	T

Figure 3–4: Generation of DNA Sequence by Regular HMM
The top portion of the figure shows a DNA sequence of length seven that is generated by a regular HMM. The 6-mers or observations that are emitted by the regular HMM to generate the given DNA sequence are given in the form of a table.

case emit 6-mers or alignment columns of 6-mers, one of the major assumptions on which the HMMs are based, that the observation O_i emitted by state S_i at the i^{th} index is only dependent on S_i and is independent of states and observations occurring at any other index is breached. Here, it can be seen that the overlapping windows are not independent, given the state the process is at. For example, we take a simple scenario where from all states all 6-mers are equally probable. If the 6-mer present at some arbitrary position i is AATTGG then the 6-mers from the alphabet {A, C, G, T}^{6} that can appear at position $i + 1$ with non-zero probability (exactly with probability $1/4$) are ATTGGA, ATTGGC, ATTGGG, ATTGGT as opposed to all the 6-mers appearing with an equal probability of $1/4^6$. To deal with this problem we tried out three different heuristic approaches:

1. Ostrich heuristic - Following the principle of ostrich effect, ”to stick one’s head in the sand and pretend there is no problem” [99], the problem can be totally ignored,

i.e. we can ignore the problem of dependence among overlapping windows despite knowing the underlying states and continue using the learning and inference algorithms designed for scenarios where observations are independent of each other given the states.

2. Parallel six heuristic - Instead of running the HMM over a sequence of overlapping 6 length windows, we can run the HMM over a sequence of non-overlapping windows. For instance, we can run the HMM on a sequence of non-overlapping windows starting at positions that are multiples of six, i.e. positions $0, 6, ...$, till the end of the sequence. But, due to this only one-sixth of the total windows are considered, and there might be a significant loss of vital information required to identify the regulatory regions. To overcome this we run six identical HMMs ($\text{HMM}_0, \text{HMM}_1, ..., \text{HMM}_5$), where HMM_j runs over windows starting at positions $\mathbb{1}\{i \mod 6 = j\}$. It is important to note that a position p is covered by multiple windows that are assumed to be generated by different HMMs. To get the posterior probability that a position p belongs to a particular state s, we average the posterior probabilities computed for all the windows covering position p. This helps us to achieve a smooth output without abrupt fluctuations.
3. Sixth root heuristic - As discussed earlier, by following the ostrich heuristic the probability of the possible 6-mers at an arbitrary position is taken to be much lower than what it actually is. As will be shown in the following chapter this underestimation of the emission probabilities has a severe negative impact on the performance of the HMMs. To scale the emission probabilities we decided to take the sixth root of the emission probabilities. This approach is mostly intuition and result driven. The

intuition behind choosing the sixth root to scale the probabilities came from the fact that every position in the DNA sequence apart from the first and last five positions is a member of six windows and contributes to the likelihood of six 6-mers. This can be seen analogous to the position being emitted six times and thus to compensate for this we decided to take the sixth root of the emission probabilities. As will be seen in the next chapter, doing this achieves significantly better performance over the ostrich heuristic.

3.4.2 Support Vector Machines

We trained support vector machine (SVM) models to solve a binary classification problem of segregating the GM12878 regulatory regions from the rest of the DNA. As discussed earlier in section 2.7, SVM is a supervised machine learning model that requires training examples from both the positive and the negative classes. While the GM12878 regulatory regions in our case serve as the positive training examples, the set of negative training examples had to be constructed by selecting for each positive example a DNA sequence having the same length, belonging to the same chromosome, and not overlapping with any of the GM12878 regulatory regions. In this manner, a negative training set having the same size as well as the same length distribution as the positive training set was constructed.

Deciding whether a DNA sequence belongs to the regulatory class or not is very similar to the text categorization problem which aims at classifying documents into different categories. Analogous to a document being seen as a sequence of words or n-grams (contiguous sequence of n words), a DNA sequence as in our case can be viewed as a sequence

of 6-mers. In text categorization, the documents are vectorized to be represented as real-valued vectors whose size is often equal to the number of words or n-grams in the corpus. Following a similar approach, the DNA sequences were vectorized to be represented as real-valued vectors in a space having the number of dimensions equal to the number of possible 6-mers, i.e. 4^6. The value assigned to component i of the vector representing a DNA sequence S is equal to the frequency of occurrence of the 6-mer associated with dimension i. This gives a high dimensional yet sparse dataset. SVMs have been shown to perform well on the text categorization problem owing to the fact that SVMs give a stable performance when the number of dimensions in the input space are high and the datasets are sparse [100]. Based on this, we expect the SVM classifier to perform well in our case as well. We trained the SVM with the k-spectrum kernel [101]. A k-spectrum kernel computes similarities between two sequences based on the frequencies of k-mers. In our case we are using 6-mers, so k = 6. This is exactly the same as the approach taken by Fletez-Brant, Dongwon Lee, Beer *et al.* for designing their tool, the kmer-SVM [58]. The length distribution of the GM12878 regulatory regions is shown below in Figure 3–5. Usually, the lengths of the regions given by the individual ChIP-seq experiments are of size 200-500 bases, but many of these regions overlap each other. These overlapping regions are merged, resulting in longer regions. Since the k-spectrum kernel is a not a length invariant kernel, large variations in the lengths of sequences can negatively affect the performance of the classifier. To avoid this, we discarded all the regions that had lengths greater than 1000 bases. Keeping only regions that have lengths less than or equal to 1000 gives us a set of regions that covers 85% of the original set of GM12878 regulatory regions. The mean length of the regions in this reduced set is 500 bases.

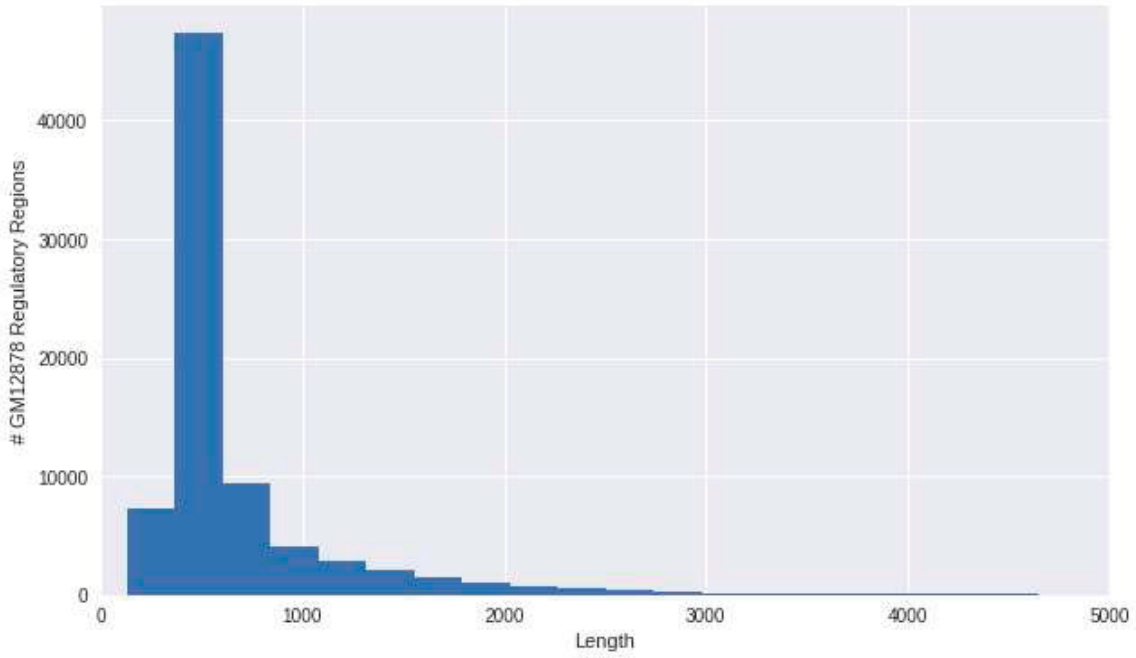

Figure 3–5: Length Distribution of GM12878 Regulatory Regions

The SVM classifier trained using the positive and negative sets described above was used to solve the sequence labelling task of identifying GM12878 regulatory DNA for the DNA sequence of an entire chromosome by adopting the following procedure:

1. Divide the DNA sequence of the chromosome represented as a sequence of 6-mers into windows of length 500 using a sliding window with a step-size of 6. This gives us a total of chromosome length/6 number of subsequences.
2. Run the trained SVM classifier on each of the subsequences. This gives a class label for every subsequence. All the positions in the sequence are covered by multiple subsequences. A position is reported as GM12878 regulatory if and only if a majority of the subsequences covering the position are classified as being GM12878 regulatory regions.

3.5 Comparing the Models in terms of Number of Parameters

The three classifiers used here, namely the phylo-HMM, the regular HMM and the SVM vary in the number of parameters that need to be estimated during learning. The greater the number of parameters, the greater the ability of the classifier to model complex processes or functions. However, given a certain amount of data, the increase in the number of parameters also increases the risk of overfitting on the training data which leads to the classifier having a high generalization error.

The SVM classifier defines the decision boundary in terms of the number of training examples. Hence, the number of parameters it needs to learn is equal to the number of training instances. However, all the training data points except the points nearest to the decision boundary on either side have the parameters associated with them set to zero as the decision boundary can be defined only in terms of the points closest to the decision boundary on either side, also called as support vectors. Thus, the number of non-zero parameters is equal to the number of support vectors. Thus, the number of parameters to be estimated is the order of $O(n)$, where n is the number of training examples. The parameters to be estimated for the regular HMM comprise of the state transition probabilities, the emission observation probabilities, and the initial state probabilities. The breakdown of the number of parameters to be estimated under each of these components is given below in Table 3–1.

Table 3–1: The Number of Parameters to be Estimated by the Regular HMM

State Transition Probabilities	Emission Observation Probabilities	Initial State Probabilities
$O(\|S\|^2)$	$O(\|\Sigma\|^k * \|S\|)$	$O(\|S\|)$

Here,

- $S-$ finite set of states
- $k-$ the length of the k-mer. In our case it is 6.
- $\Sigma-$ alphabet set, which in our case is $\{A, C, G, T\}$

The phylo-HMM needs the same number of parameters to be estimated for the state transition probabilities and for the initial state probabilities as in case of the regular HMM. However, the number of parameters needed to evaluate the likelihood of an alignment column given the state significantly exceeds the number of emission observation probabilities to be estimated in case of the regular HMM. Recall from Equation 2.14 that in order to calculate the likelihood of a column X_i given the process is in some arbitrary state s we need to estimate:

- Probability of character a occurring at the root conditioned on state s : $\mathsf{P}[x_r = a|s], \forall a \in \Sigma$ and $s \in S$.
- Probability of character a being substituted by character b along the edge (u, v) when the process is in state s: $\mathsf{P}[x_v = b|x_u = a, s], \forall a, b \in \Sigma$, $s \in S$ and $(u, v) \in E_\tau$.

The number of parameters to be estimated by phylo-HMM for each of the components is given below:

Table 3–2: The Number of Parameters to be Estimated by the Phylo-HMM

State Transition Probabilities	$\mathsf{P}[x_r = a\|s]$	$\mathsf{P}[x_v = b\|x_u = a, s]$	Initial State Probabilities
$O(\|S\|^2)$	$O(\|\Sigma\|^k * \|S\|)$	$O(\|\Sigma\|^{2k} * \|E_\tau\| * \|S\|)$	$O(\|S\|)$

Here,

- $S-$ finite set of states.
- $k-$ the length of the k-mer.
- $\Sigma-$ alphabet set, which in our case is $\{A, C, G, T\}$.

- E_τ– set of edges in the phylogenetic tree τ.

From, the discussion above we can clearly see that the phylo-HMM is capable of modeling much complex processes than the regular HMM and the SVM, while also being more vulnerable to overfitting.

CHAPTER 4
Result Analysis

In this chapter, intermediate as well as final results are discussed and analyzed.

4.1 Hidden Markov Models

The regular HMMs as well phylo-HMMs were trained on data from chromosomes 1, 5, 13, 18, 19, 20, and 21; while their performance was tested on chromosome 8. The choice of chromosomes for training and testing was uniformly random.

4.1.1 Selection of Heuristics

To check the feasibility of the three heuristics discussed in section 3.4.1, namely, the ostrich heuristic, the parallel six heuristic, and the sixth root heuristic; we tested the performance of the two-state regular HMM on chromosome 7. Note that chromosome 7 was used as cross-validation data as we did not want the heuristic selection to be based on the test data of chromosome 8. A portion of the most likely state sequence produced by the Viterbi algorithm in all the three cases along with true state labels is shown in Figure 4–1. From this, we can see that the predictions for the parallel six and the sixth root versions are similar to each other and correspond to the true labels on many occasions. On the other hand, the predictions made by the HMM using the ostrich heuristic is seen to have very frequent transitions between the two states, leading to a large number of false predictions (both false positives and false negatives). This can be owed to the fact that when using the ostrich heuristic, the emission probabilities are highly underestimated as discussed in section 3.4.1. This causes the estimated emission probabilities to be much

lower in value as compared to the transition probabilities, and this leads to frequent state transitions. Having looked at this result, we became aware of the flaw in using the ostrich heuristic and decided to discard it from future experiments while the other two heuristics were tested further with different structures.

4.1.2 Comparing the Parallel Six and the Sixth Root Heuristics

The performance for regular as well as phylo-HMMs with different structures using the parallel six and the sixth root heuristics was reported on chromosome 8 in terms of the receiver operating characteristic (ROC) curve and the precision-recall (PR) curve. The ROC curve defined for a two-class (positive class and negative class) classifier is a plot of *sensitivity* also called as recall or true positive rate (TPR) against 1− *specificity* also known as the false positive rate (FPR). The expressions for sensitivity and specificity in terms of true positives (TP), false positives(FP), false negatives (FN), and true negatives (TN) are given below in equations 4.1 and 4.2. Sensitivity is the fraction of total positive observations classified correctly, and specificity is the fraction of the negative observations classified correctly.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.1}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{4.2}$$

A probabilistic binary classifier outputs for a given observation the probability of belonging to the positive class versus belonging to the negative class. The decision of which label should be assigned to the given observation depends on whether the output probability is above or below the decided threshold. Every point on the ROC curve gives the sensitivity and the 1− specificity pair corresponding to a particular decision threshold.

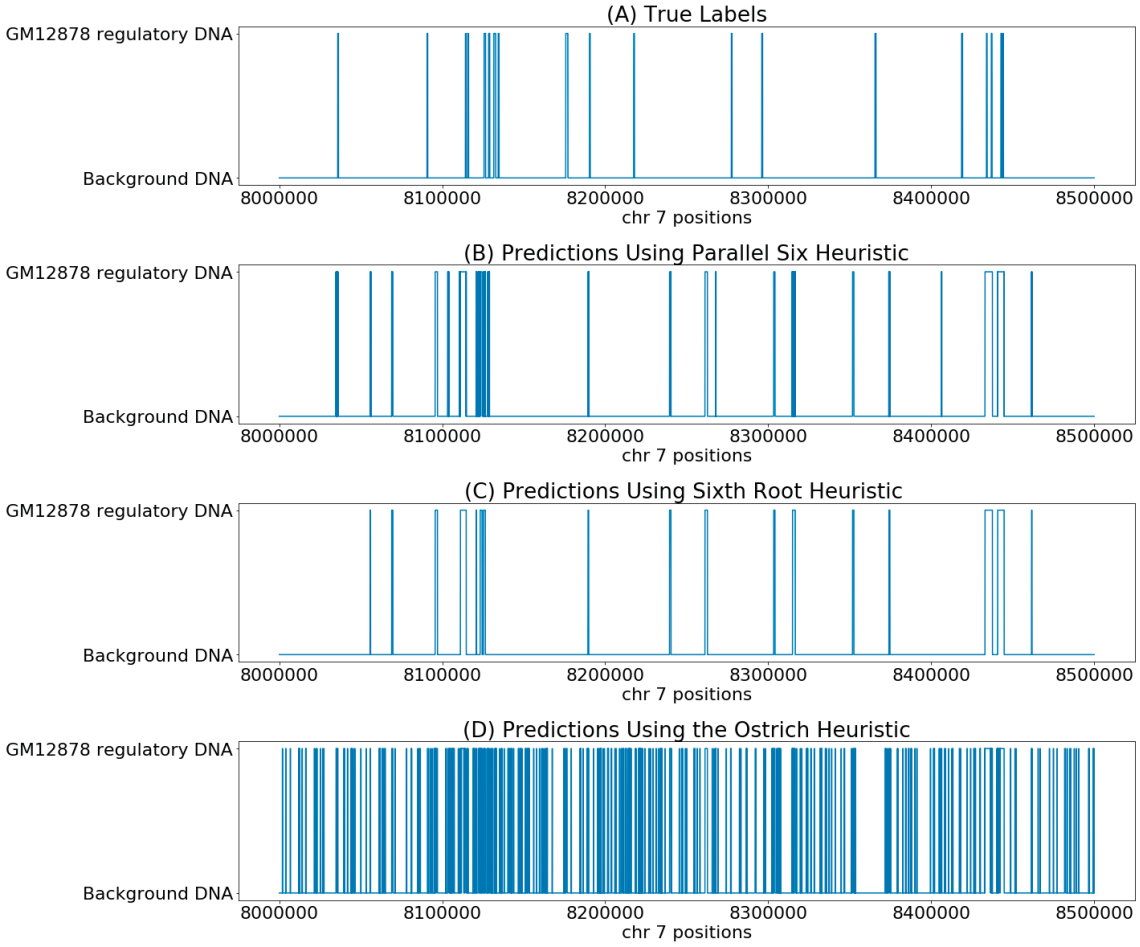

Figure 4–1: Comparison of the Different Heuristics.
(A) shows the true labels for the portion of chromosome 7 (chr7). (B) shows the predictions recorded by running the Viterbi algorithm for the two-state regular HMM using the parallel six heuristic. (C) shows the Viterbi output of the two-state regular HMM using the sixth root heuristic. (D) shows the output of the Viterbi algorithm for the two-state regular HMM using the ostrich heuristic.

Similarly, the PR curve represents the trade-off between *precision* and recall for different decision thresholds. Precision is the measure of how accurate is the classifier when it predicts an observation to be positive. The expression for precision is given below:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.3}$$

To report the performance for the three-state and the four-state HMMs in terms of the ROC and the PR curves the view of a one-versus-all classifier was adopted where all the observations predicted to belong to any of the classes other than the GM12878 regulatory class are seen as the same.

The output of the forward-backward algorithm was used to plot the ROC curve, and the area under the curve (AUC) was calculated for every case. The AUC value for a given ROC curve is a measure of the ability of the classifier to demarcate between the two classes. An area of 0.5 corresponds to a random classifier while the area of 1 corresponds to a perfect classifier. Figure 4–2 shows these ROC curves along with the AUC values. There are many interesting observations that one can make from these plots. In case of the regular HMM for all three structures, the curves for both the heuristics almost overlap each other throughout and looking at the AUC values one cannot decide on picking one heuristic against the other. Coming to the phylo-HMM results, for the two-state case, the observations are very similar to those in case of the regular HMM. However, for the three-state and the four-state phylo-HMMs, the parallel six heuristic beats the performance of the sixth root heuristic, but there is a significant drop in performance for both heuristics when compared to the two-state variant.

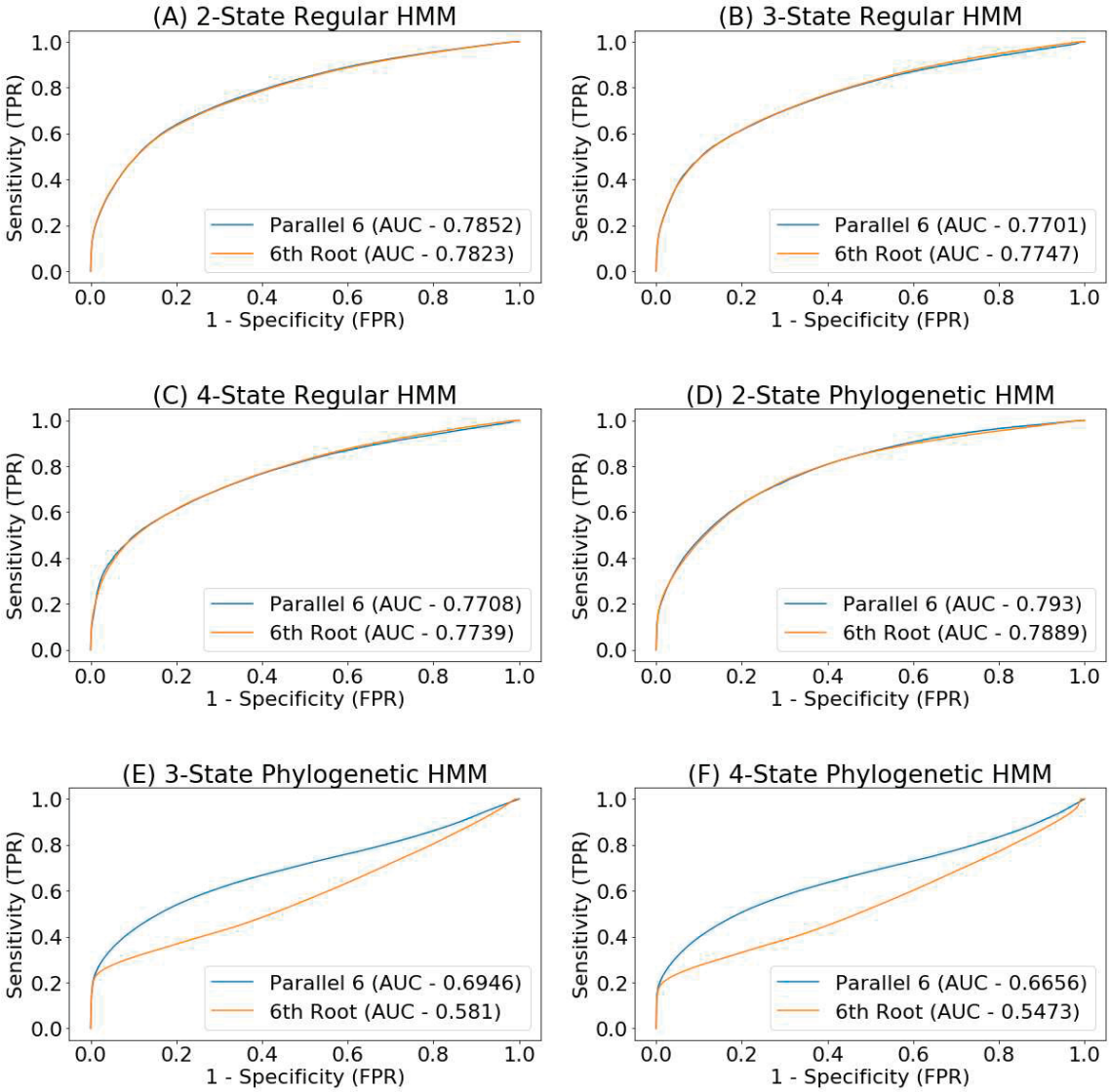

Figure 4–2: Comparing the Parallel Six and the Sixth Root Heuristics in Terms of ROC Curves

Next, the PR curves were plotted which can be seen in Figure 4–3. The story portrayed by the PR curves is also the same as the one depicted by the ROC curves. In case of the 2-state and the 3-state regular HMM, the two curves significantly overlap each other, while in the 4-state regular HMM and the 2-state phylogenetic HMM, the curves cross each other at multiple locations signifying that no heuristic performs better than the other throughout the entire range of recall.

4.1.3 Comparison within Limited False Positive Rate

As mentioned earlier in section 3.1, the GM12878 regulatory regions cover only $\sim$ 1.7% of the entire genome leading to an imbalanced ratio $<$ 1:50 between the GM12878 regulatory and the rest of the DNA. Thus, where for a balanced dataset the false positive rate (FPR) of 20% is seen as good the same FPR in our case will imply that the number of false positive predictions exceeds the total number of positive observations by more than ten folds. This is unacceptable for practical purposes and for the classifier to be useful, the amount of false positives need to be limited. We limited the FPR to be at most 5% and observed the behavior of the ROC curves corresponding to the HMMs with different structures and using different heuristics. Figure 4–4 shows the behavior of the ROC curves within the acceptable FPR range. We see that no single curve is above all the other curves for the entire range of acceptable FPR. The curves cross each other at one or more locations. Two ROC curves crossing each other imply that one of the classifiers is better than the other for the range of FPR values where its ROC curve lies above, while the other classifier is better in regions outside this range. From Figure 4–4 we can see drastic changes in the relative performance of the classifiers as the FPR values change. For instance we can see that the performance of the parallel six four-state regular HMM

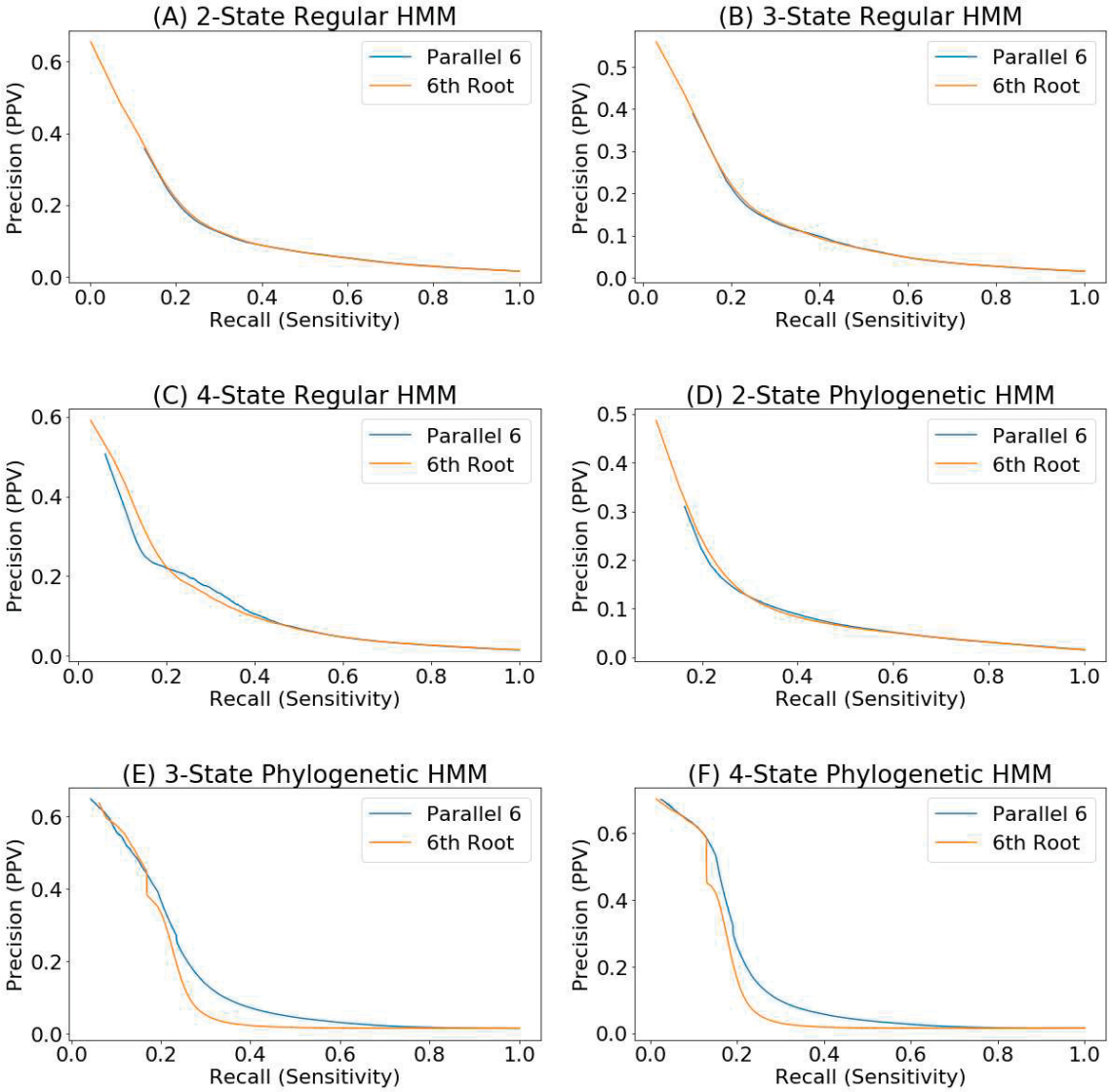

Figure 4–3: Comparing the Parallel Six and the Sixth Root Heuristics Using PR Curves

is the worst among all the classifiers for FPR < 1%, but then its performance improves steeply to become the best beyond the FPR value of just below 2%. On the other hand, the parallel six three-state phylo-HMM, the parallel six four-state phylo-HMM, and the sixth root three-state phylo-HMM start as the top three performers before falling behind just before the 2% FPR mark. Thus, even by looking at only the acceptable range of FPR values we cannot decide upon which structure, heuristic, and variant of the HMM to use over the entire range of acceptable FPR values. One other critical observation we can make is that even though we don't see an increase in the AUC values with increase in state space, increased state space does have positive effects in certain intervals of the acceptable FPR range.

4.1.4 Combined Performance of Regular and Phylogenetic Hidden Markov Models

As seen in the previous section different HMM classifiers perform best at different intervals of the FPR range and no single classifier performs the best throughout the acceptable range. We decided to combine the predictions given by the regular and the phylo-HMM to see if their combine performance could beat their individual performances and perform better than the other HMM classifiers for the entire range of FPR.

The results of the regular and phylo-HMM are combined by taking the intersection of the predictions made by them, i.e. reporting only those windows as positive that were predicted as positive by both. To plot the ROC curve of the combined performance we divided the FPR range spanning from 0 to 1 into hundred intervals. Next, from each of these intervals we selected one point from each of the two ROC curves corresponding to the regular and the phylo-HMM classifiers. The decision thresholds at which these points were plotted were looked up. Using these decision thresholds we got the sets of

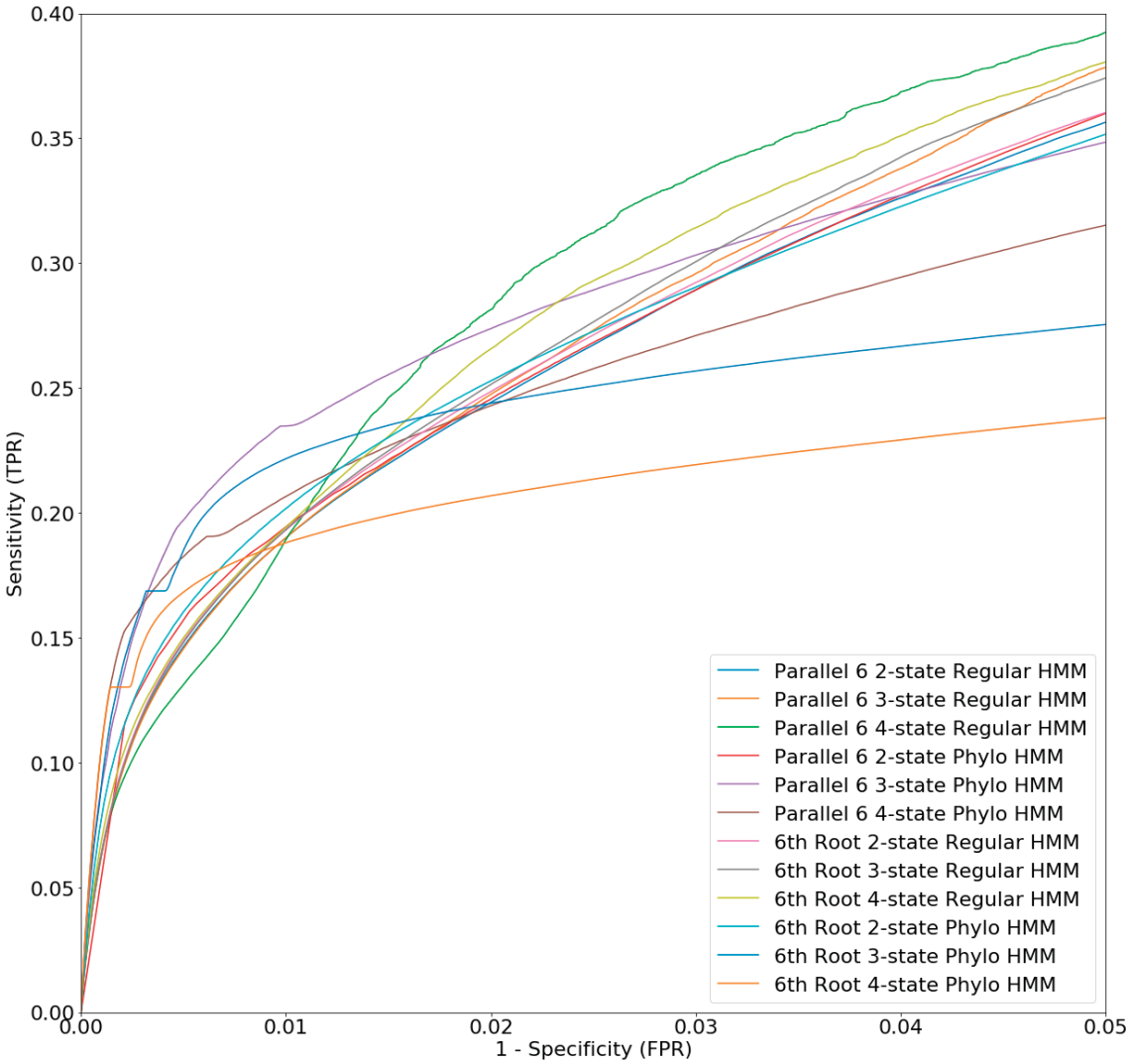

Figure 4–4: Comparing Different HMM Classifiers within Limited False Positive Rate Using ROC Curves

windows predicted to be GM12878 regulatory in each case. Taking the intersection of these predictions gave us a new point on the combined ROC curve. It is important to note that this is a specificity increasing and sensitivity reducing technique. Let us consider a and b to be the two points on the two ROC curves. Let TP_a, FP_a, specificity_a, and sensitivity_a denote the set of TP predictions, set of FP predictions, value of specificity, and value of sensitivity associated with point a. Similarly, let us denote these quantities by TP_b, FP_b, specificity_b, and sensitivity_b for point b, and by TP_{ab}, FP_{ab}, specificity_{ab}, and sensitivity_{ab} for the point achieved by intersecting the predictions corresponding to a and b.

$$
\begin{aligned}
&\text{TP}_{ab} \leftarrow \text{TP}_a \bigcap \text{TP}_b \\
&\Rightarrow |\text{TP}_{ab}| \leq \min(|\text{TP}_a|, |\text{TP}_b|) \\
&\Rightarrow \text{sensitivity}_{ab} \leq \min(\text{sensitivity}_a, \text{sensitivity}_b) \\
&\text{Similarly, } \text{FP}_{ab} \leftarrow \text{FP}_a \bigcap \text{FP}_b \\
&\Rightarrow |\text{FP}_{ab}| \leq \min(|\text{FP}_a|, |\text{FP}_b|) \\
&\Rightarrow \text{specificity}_{ab} \geq \min(\text{specificity}_a, \text{specificity}_b) \\
&\delta_{\text{sensitivity}} = |\min(\text{sensitivity}_a, \text{sensitivity}_b) - \text{sensitivity}_{ab}| \\
&\delta_{\text{specificity}} = |\min(\text{specificity}_a, \text{specificity}_b) - \text{specificity}_{ab}|
\end{aligned}
$$

The technique of intersecting predictons will be beneficial if $\frac{\delta_{\text{sensitivity}}}{\delta_{\text{specificity}}} < 1$,i.e. if there is a large gain in specificity for a small loss in sensitivity. The smaller the ratio, the larger the benefits. For the ratio to be small, it is necessary that both the classifiers have a large amount of true positive predictions in common, but do not have a large number of false positives in common. By randomly sampling the predictions we observed that there were many instances where the regular HMM made false positive predictions and

the phylo-HMM did not. This happened typically in regions where the sequence readout was indicative of GM12878 regulatory function, but the regions were not conserved. An example of such a prediction is shown in Figure 4–5. Figure 4–5 shows a view of the UCSC Genome Browser (http://genome.ucsc.edu/) [69]. One can see that the regular two-state HMM assigns positive scores to two poorly conserved regions that are not functional in GM12878 cell line, while the same region is given negative scores by the phylo-HMM. On the other hand, we also found a good number of regions that were predicted as false positives by the phylo-HMM and not by the regular HMM. These regions were ones that were generally well conserved, but their sequence readout was not convincing enough for the regular HMM to assign them a high posterior probability of being GM12878 regulatory. The example of such a prediction can be seen in Figure 4–6. From Figure 4–6 we can see that a highly conserved non-functional in GM12878 cell line DNA region is scored positively by the phylo-HMM but not by the regular HMM. One interesting thing to note is the increase in the regulatory-score assigned by the regular HMM. Even though this increase does not take the score to be positive, it takes it very close to being positive. This is indicative of the fact that even the sequence readout of this region has information that is indicative of GM12878 regulatory function and hence, the predictions made by the phylo-HMM are not only driven by sequence conservation. It is important to note that this discussion on the nature of the false positive predictions made by the regular HMM and phylo-HMM are based on the predictions that we randomly sampled and manually inspected by visualizing them in the UCSC Genome Browser (http://genome.ucsc.edu/) [69] and since we could analyze only a fraction of the regions due to time limitations, we

cannot completely rule out the possibility of existence of regions for which the nature of false positive predictions may be different.

We combined the performance of the regular and the phylo-HMM keeping the structure and the heuristic fixed. The ROC curves denoting the combined performance, along with the ROC curves for the regular and phylo-HMM for each combination of structure and heuristic are shown in Figure 4–7. In all the cases we can see that the combined ROC curve indicates performance better than at least one of the two classifiers throughout the range of FPR, and for the two-state parallel six and the two-state sixth root HMMs the combined ROC curve shows better performance than both the classifiers. Also, for the three-state parallel six and the four-state parallel six the combined ROC curve is above the other two curves for lower values of FPR before it is overtaken by the curve corresponding to the regular HMM. Next, we compare the performance of all the variants seen so far with the FPR limited to 5%. Figure 4–8 shows this comparison. We see that the problem of no classifier dominating for the entire range does not show up here. The top three curves showing the combined performances of the regular and phylo-HMMs in case of the parallel six two-state HMM, the sixth root two-state HMM, and the parallel six three-state HMM are above all the other ROC curves for the entire range of acceptable FPR, with combination of parallel six two-state regular and phylo-HMMs performing the best. Hence, now we can choose a single classifier for the entire range of acceptable FPR.

4.2 Support Vector Machines

We trained the support vector machine (SVM) classifier using all the regions from the reduced positive and negative training sets described in section 3.4.2, except the regions from chromosome 8 and chromosome 2. The SVM we used was a soft-margin SVM and

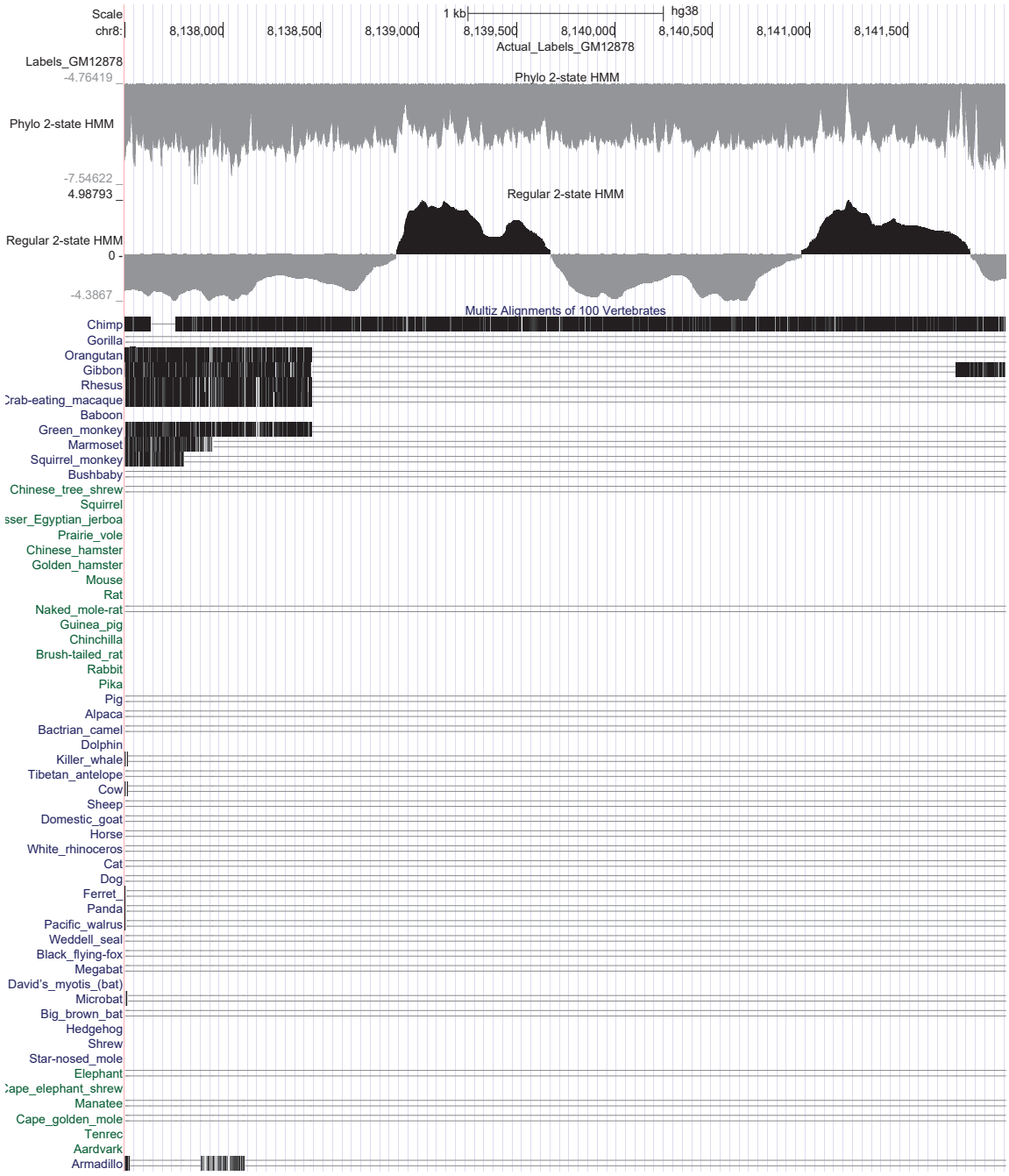

Figure 4–5: False Positive Predictions made by the Regular HMM

Figure 4–5: The different tracks starting from the top are:

1. The Labels_GM12878 track indicating the presence of GM12878 regulatory DNA. The GM12878 regulatory regions are shown in black and for all the other regions there is no fill at all.
2. The Phylo 2-state HMM track shows the output of the forward-backward algorithm for the 2-state phylo-HMM. The posterior probability that the six-length window starting at the particular position belongs to the GM12878 regulatory set is not plotted directly. The posterior probability is used to obtain a regulatory-score that when plotted can be visually more informative. The computation of the regulatory-score is given by equation 4.4, where p is the posterior probability of the six-length window belonging to the GM12878 regulatory set which is rounded off to the ninth decimal place.

$$\text{Regulatory-Score} = \begin{cases} \log_{10}(p) & \text{if } 0 < p < 0.5 \\ -\log_{10}(1-p) & \text{if } 0.5 \leq p < 1 \\ -10 & \text{if } p = 0 \\ +10 & \text{if } p = 1 \end{cases} \tag{4.4}$$

3. The Regular 2-state HMM track shows the regulatory-scores obtained using the regular 2-state HMM.
4. The MULTIZ alignments track shows the MULTIZ 100way alignment. Here, we have customized this track to show information only from the 58 present day species that we are using. The fill pattern used by the UCSC Genome Browser is that the darker the shade, the more is the sequence conserved, with the absence of fill denoting gaps.

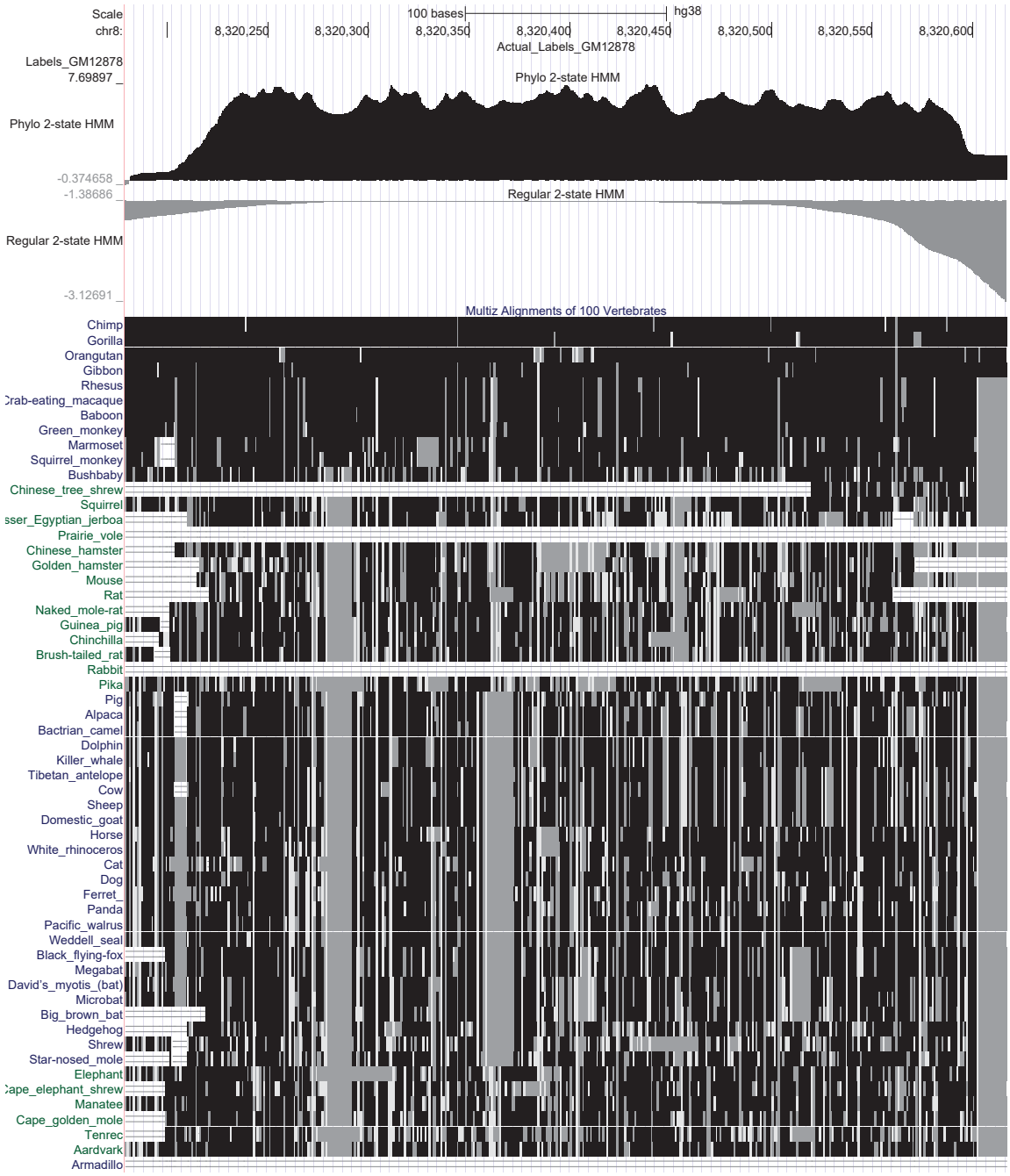

Figure 4–6: False Positive Predictions made by the Phylogenetic HMM

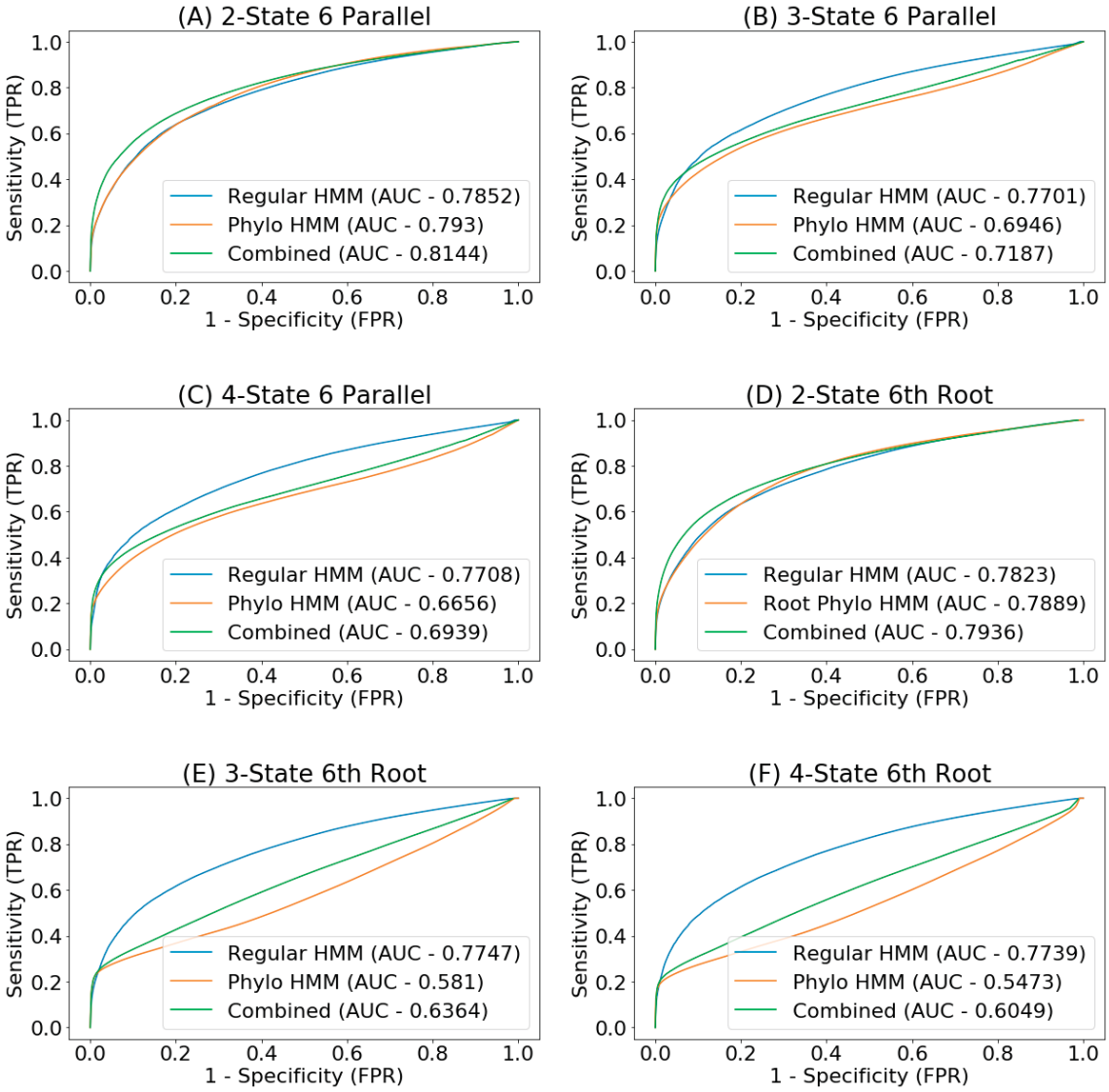

Figure 4–7: Combining the Regular and the Phylogenetic HMMs

hence, the parameter C controlling the degree of penalization of the misclassifications had to be chosen. This was done by following a stratified 5-fold cross-validation (CV) strategy. In this the entire training set is randomly divided into 5 equal size subsets, each containing the same proportion of negative and positive examples. Then one of the 5

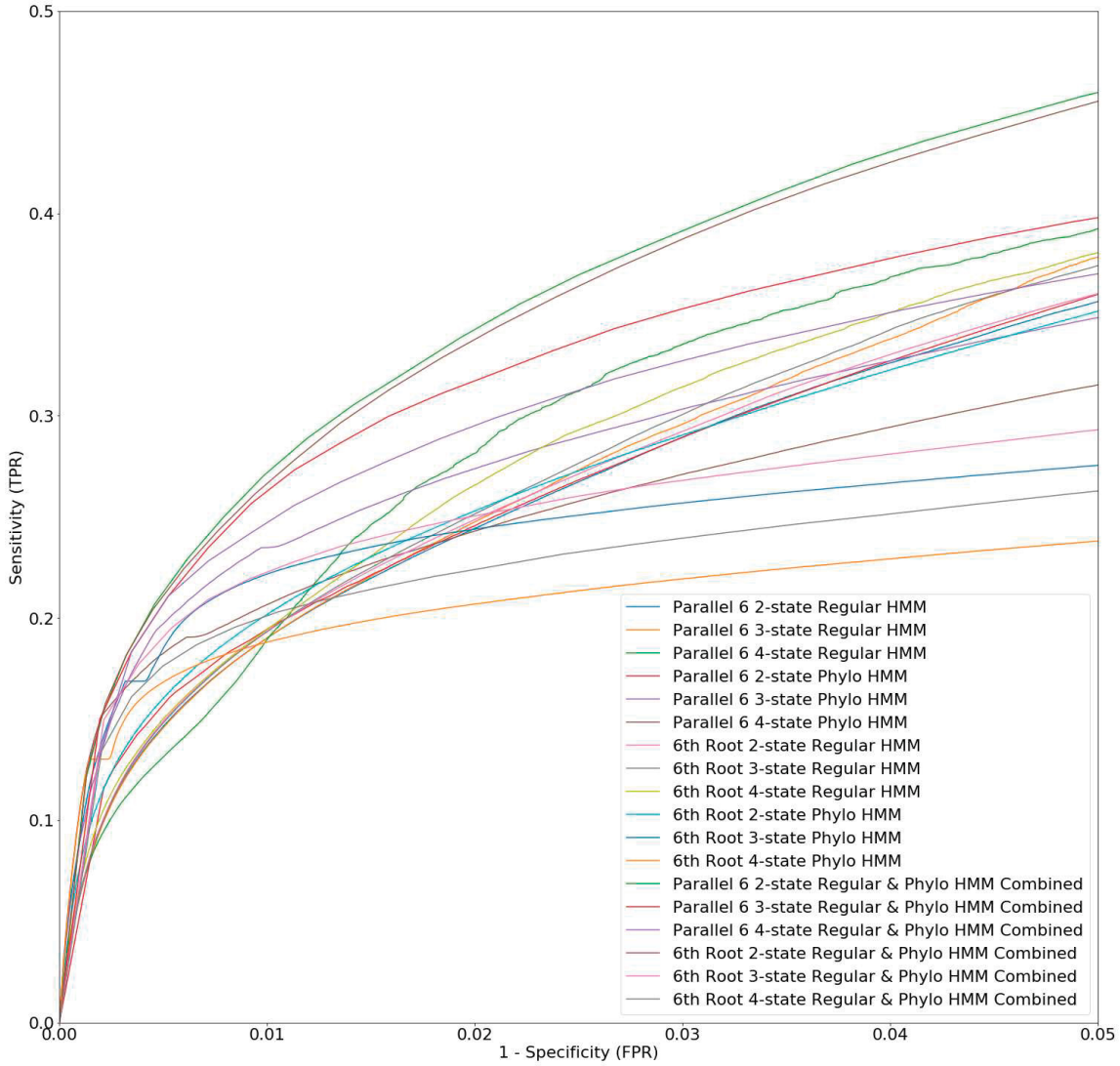

Figure 4–8: ROC Curves for all HMM Classifiers along with their Combinations within Limited False Positive Rate

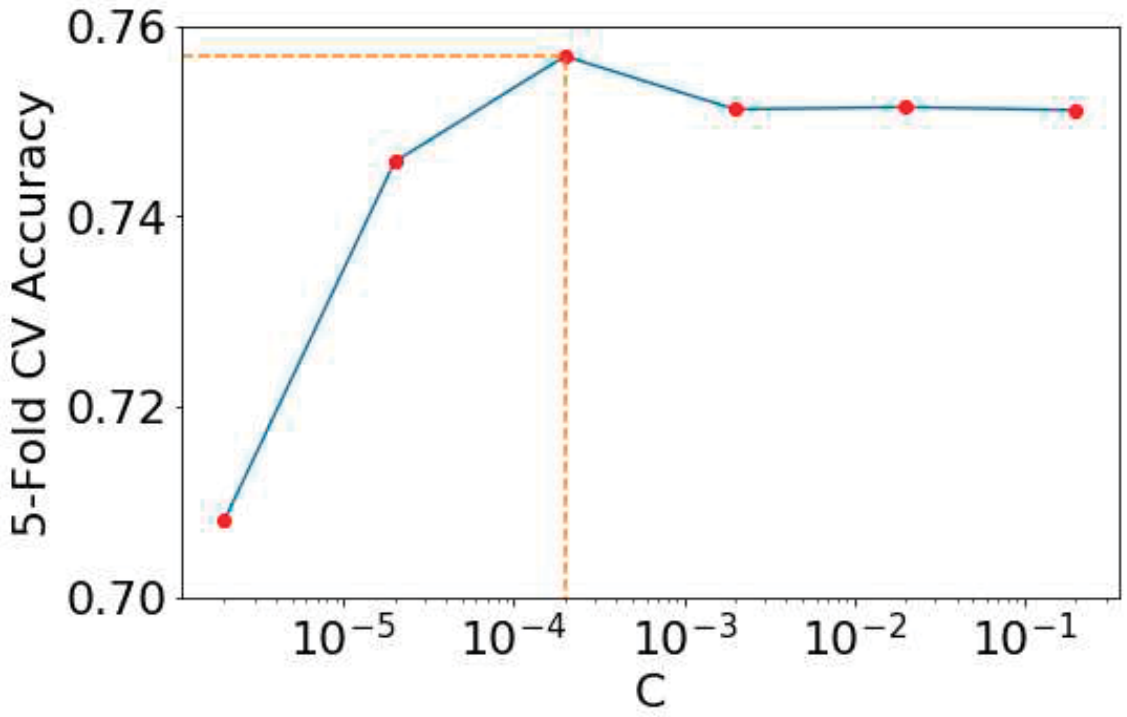

Figure 4–9: Five-Fold Cross Validation Accuracy for the SVM Classifier versus Penalty Parameter C

Note: The graph has been plotted in log scale for better resolution

subsets is retained as the validation set to test the classifier and the remaining four are used for training the model. This is repeated for five times using a different subset as the validation set every time. The test results on all of the subsets are then averaged and presented as the cross-validation performance. The 5-fold CV was performed using the different values of C, and the value that gave the best result was chosen. The 5-fold CV accuracies for different values of C are shown below in Figure 4–9. The value $C = 2\times10^{-4}$ gave the best 5-fold CV accuracy of 75.68%.

Next, we ran the trained SVM classifier on the whole of chromosome 8 using the sliding window and labelling of positions based on the labels of the overlapping subsequences approach, as mentioned in section 3.4.2. Since SVM is not a probabilistic classifier we do not get a ROC curve, but just a single pair of specificity and sensitivity values. Our SVM classifier gave us a specificity of 74.52% and a sensitivity of 76.61%. As discussed earlier in section 4.1.3, due to the large class imbalance in our case, a classifier with such a high false positive rate is of little practical importance. However, it may prove to be useful when used in combination with other classifiers.

4.2.1 Combined Performance of Support Vector Machines and Hidden Markov Models

The performance of an HMM was reported as a ROC curve, i.e. pairs of values of sensitivity and 1-specificity or false positive rate (FPR), whereas the performance of our SVM classifier is reported as a single sensitivity, FPR pair. A ROC curve corresponding to the combined performance can be drawn by plotting the sensitivity and FPR values achieved by intersecting the predictions made by the HMM at different FPRs with the predictions made by the SVM. Figure 4–10 shows the performance achieved by intersecting the predictions given by the combination of the parallel six two-state regular and phylogenetic HMM, which was the best classifier among all the HMM classifiers, with the predictions of the SVM classifier. One can see that there is a slight increase in performance by using the predictions given by the SVM classifier. The following facts can explain this slight increase. One, due to the high true positive rate of the SVM classifier, almost every true positive prediction made the combined HMM classifier was predicted to be positive by the SVM classifier. Second, almost all the false positive predictions made by the regular HMM which were already discarded on taking the intersection with the predictions

of the phylo-HMM, match with the false positive predictions made by the SVM classifier, causing them also to be discarded. Recall that these are regions where the sequence readout of the region is indicative of GM12878 regulatory function, but the region itself is not well conserved. Figure 4–11 shows a screenshot of the UCSC Genome Browser (http://genome.ucsc.edu) showing a portion of chromosome 8. Here apart from the tracks mentioned in section 4.1.4, a track showing SVM predictions has been added. It can be seen that the SVM classifier falsely predicts five regions, four of which are predicted to be positive by the regular 2-state HMM also. All of the predicted regions are not well conserved and are predicted correctly to be negative by the phylogenetic HMM.

4.3 Analysis of False Positive Predictions

It is important to analyze the composition of the false positive predictions made by the classifier to better understand the mistakes the classifier is making and how it can be improved in the future. The composition of false positive predictions obtained after intersecting the predictions made by the SVM classifier with the predictions made by the combination of the parallel six two-state regular and phylo-HMMs was analyzed. We limited the FPR at 5% and counted the number of false positive predictions that belonged to the following sets of regions:

1. The non-GM12878 regulatory set.
2. The set of DNase-I hypersensitive sites. This set was built by combining the results from DNase-I hypersensitivity assays on 95 cell types by the John Stamatoyannapoulos lab at the University of Washington, as part of the ENCODE project first production phase [102]. It was downloaded from the ENCODE data repository at the UCSC Genome Browser (http://genome.ucsc.edu/) [69, 94, 70]. Then

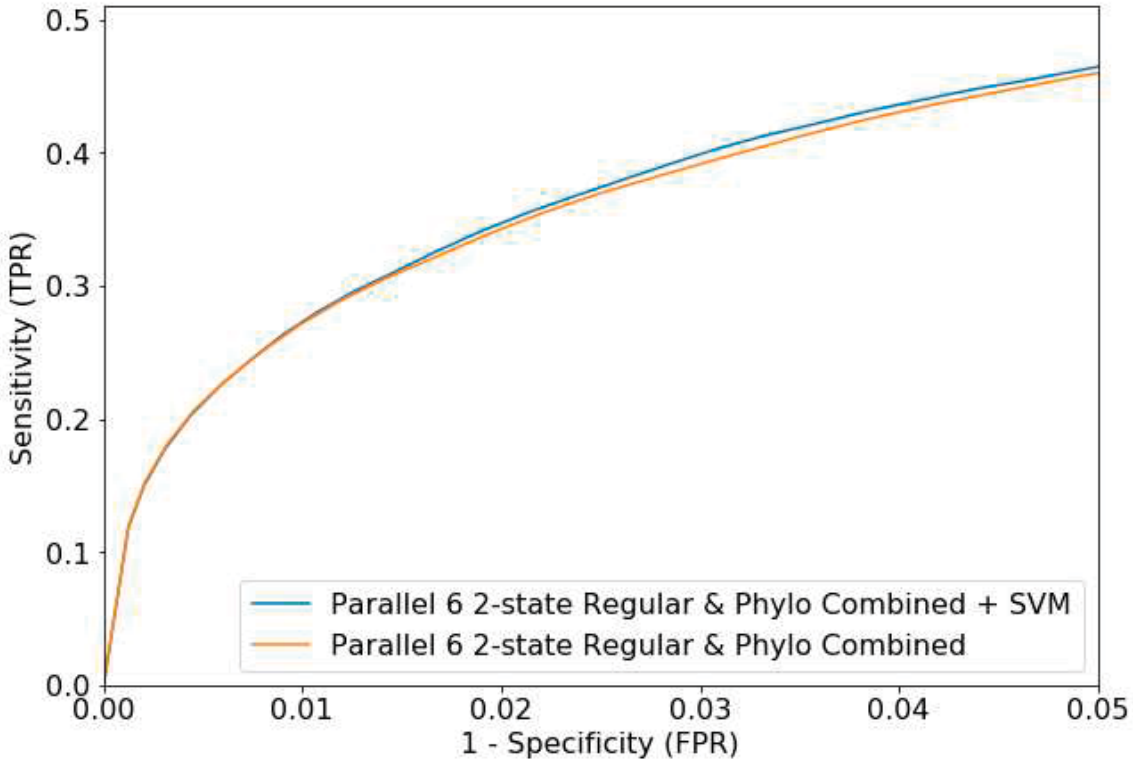

Figure 4–10: Intersecting the Predictions of the SVM Classifier with the Predictions of the Combination of Regular and Phylogenetic HMMs

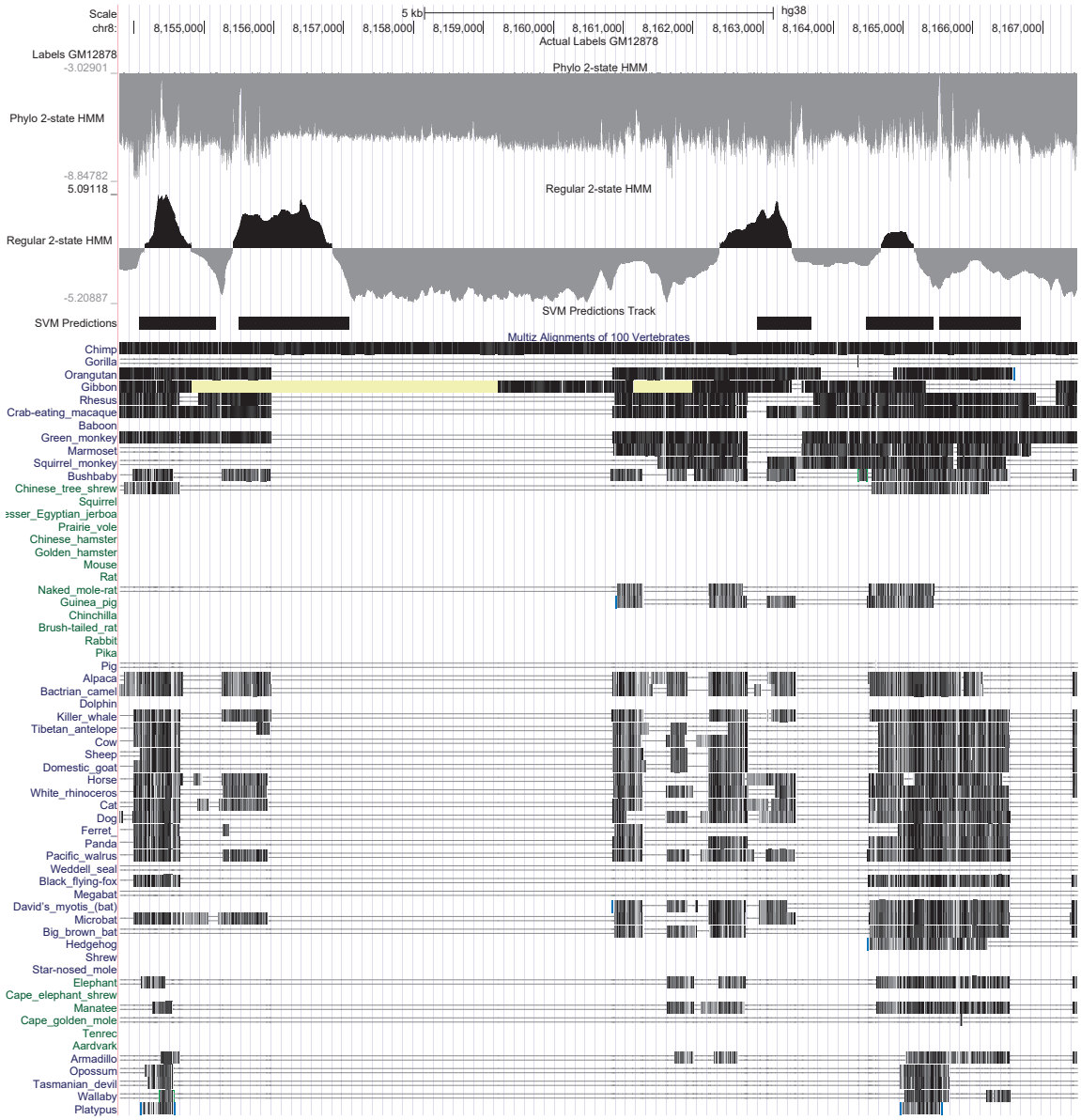

Figure 4–11: False Positive Predictions made by the SVM Classifier

this set was divided into two subsets, one containing DNase-I hypersensitive sites for the GM1878 lymphoblastoid cell line and the other containing all the hypersensitive sites other than the ones for the GM12878 lymphoblastoid cell line. For simplicity, we will refer the two subsets as GM12878 DNase-I hypersensitive sites and non-GM12878 DNase-I hypersensitive sites. The DNase-I hypersensitive sites indicate portions of open chromatin that are sensitive to digestion by the DNase enzyme and provide a good approximation of the regulatory regions as regulatory regions in general, tend to be DNase-sensitive [103]. Even though both ChIP-seq and DNase-I hypersensitivity assays are used for identifying regulatory regions, the results of both the experiments do not match completely as both have their limitations. For instance, DNase-I hypersensitivity assay is better at capturing promoters than enhancers as the former are more DNase-I hypersensitive [103] and the ChIP-seq experiments are limited by the TFs for which the antibodies are available.

The composition of the false positive predictions is presented below in table 4.3. We see that almost 53% of the false positive predictions are ones that were designated to be regulatory in some other cell type by at least one of the two experiments. Next, we checked the composition of the false positive predictions made by the combination of the parallel six three-state phylo and regular HMMs to see if adding the third state to model the non-GM12878 regulatory DNA was beneficial or not. This composition is summarized in table 4.3. It can be seen that the overall fraction of false positives that were recognized as non-GM12878 regulatory DNA by at least one of the two experiments goes to down to $\sim$ 41.87%. This drop of $\sim$ 12% from the two-state case is indicative that adding the extra state does increase the discriminatory power of the classifier to differentiate between regulatory

regions from different cell types, but only to a certain extent. This can be attributed to the fact that the underlying mechanism for cell type specificity is that of chromatin structure remodelling, and knowledge about chromatin structure cannot be overcome with DNA sequence information.

Table 4–1: Composition of False Positives Achieved after Intersecting the Predictions of the SVM and the Combination of the Parallel Six Two-State Regular and phylo-HMMs

Class	% Composition
non-GM12878 regulatory regions	35.47%
GM12878 DNase-I hypersensitive sites	0.33%
non-GM12878 DNase-I hypersensitive sites	17.56%

Table 4–2: Composition of False Positives Achieved after Intersecting the Predictions Made by the Parallel Six Three-State Phylogenetic and Regular HMMs

Class	% Composition
non-GM12878 regulatory regions	27.32%
GM12878 DNase-I hypersensitive sites	0.28%
non-GM12878 DNase-I hypersensitive sites	14.55%

4.4 Analysis of Over-Represented K-mers

The k-mers, where k = 6, which are much more probable in the GM12878 regulatory state than the other states were looked up using the emission probability tables and were compared against motifs of known TFs to see if our classifier had implicitly identified the motifs or portions of motifs. Using the emission probability table of the regular HMM, the 6-mers were ranked according to the odds ratio of being emitted from the GM12878 regulatory state versus being emitted from any other state. The expression for the odds ratio is given by equation 4.5. Following this top 50 6-mers with the highest odds ratios

were selected to be analyzed.

$$R_o = \frac{\mathsf{P}[o|S_{\text{GM12878}}]}{\mathsf{P}[o|s \in S \setminus \{S_{\text{GM12878}}\}]} \tag{4.5}$$

Here,

- o is the 6-mer.
- S is the associated state space.
- S_{GM12878} is the state that models the GM12878 regulatory DNA.
- $S \setminus \{S_{\text{GM12878}}\}$ denotes set of all states except S_{GM12878}.

Similarly, top 50 over-represented 6-mers for the phylo-HMM were also recognized. To find 6-mers that were more probable and also more conserved in the GM12878 regulatory state, we ranked the 6-mers according to the ratio given by equation 4.6.

$$R_o = \frac{\mathsf{P}[x_r = o|S_{\text{GM12878}}] \prod_{(u,v)\in E_\tau} \mathsf{P}[x_v = o|x_u = o, S_{\text{GM12878}}]}{\mathsf{P}[x_r = o|S \setminus \{S_{\text{GM12878}}\}] \prod_{(u,v)\in E_\tau} \mathsf{P}[x_v = o|x_u = o, S \setminus \{S_{\text{GM12878}}\}]} \tag{4.6}$$

Here,

- x_u is the notation used to represent the character present at node u.
- E_τ is the set of edges present in the tree τ.
- r is root node.
- All the other notations are same as equation 4.5.

The over-represented 6-mers found were then compared to known motifs using the Tomtom motif comparison tool [104]. Tomtom compares the queried sequence (in our case, the 6-mer) against a database of known motifs and produces matches having E-values beyond a certain threshold. We checked our over-represented 6-mers against the Homo sapiens Comprehensive Model Collection v11 core (HOCOMOCO v11 CORE) database

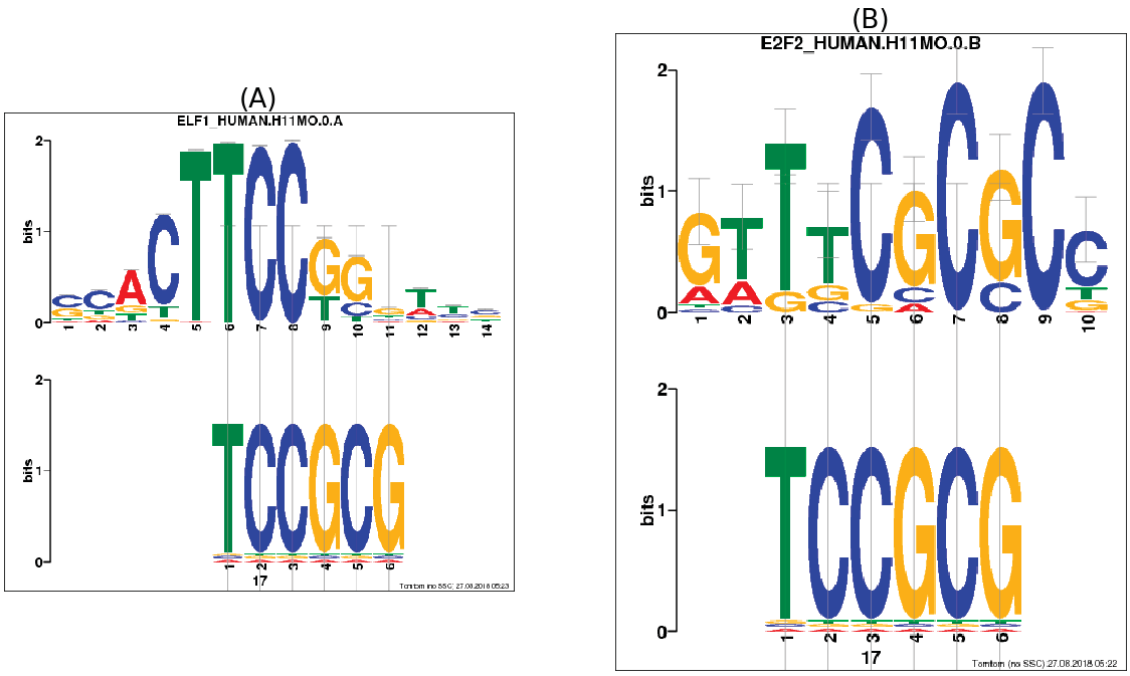

Figure 4–12: Matching of an Over-represented 6-mer to Known Motifs
(A) shows the match between the over-represented 6-mer TCCGCG and the motif for the transcription factor ELF1 [106] found in the HOCOMOCO v11 CORE database [105]. The match has an *E*-value of 5.41 and a *p*-value of 1.35e-02. (B) shows the match between the over-represented 6-mer TCCGCG and the motif for transcription factor E2F2 [107] found in the HOCOMOCO v11 CORE database [105]. The match has an *E*-value of 3.54 and a *p*-value of 8.80e-03. Both (A) and (B) were presented as outputs by the Tomtom motif comparison tool [104].

[105], which contains binding motifs for TFs binding human DNA. A total of 91 matches were found for the *E*-value threshold of 10. Figure 4–12 shows two of the matches for one of the over-represented 6-mer. The complete list of the top fifty over-represented 6-mers recognized by the regular and the phylogenetic HMM along with the ids of the motifs they matched to from the HOCOMOCO v11 CORE database can be found in Appendices A and B.

CHAPTER 5
Conclusion and Future Work

5.1 Summary of Contributions

This thesis aimed by building machine learning models to computationally identify cell type specific regulatory regions within the genome of some concerned species using the sequence information from the genomes of the concerned species and its orthologous species.

The problem of identifying regulatory regions is worthy enough for the purpose of better understanding patterns and mechanisms of gene expression regulation, and also for the purpose of disease diagnosis and drug discovery as many diseases are shown to be linked with mutations occurring in the regulatory regions. While experimental approaches do exist to identify regulatory regions, they have limitations in the form of cost, time, and availability of specific chemicals needed to carry out an experiment. These limitations make the computational approaches an invaluable asset to the researchers.

In this work we trained machine learning models to identify regulatory regions in the GM12878 lymphoblastoid cell line in humans. We leveraged the information about sequence readout and sequence evolution due to two important properties of regulatory regions. First, regulatory sequences contain DNA segments usually 5-15 bases long called transcription factor binding sites to which transcription factors bind, making the sequence composition of the regulatory regions different from that of non-functional DNA. Second,

regulatory regions are more likely to be conserved as compared to the non-functional regions, as mutations in the functional DNA more often than not have negative impact on an organism's survival and reproduction rates.

We trained the support vector machine (SVM) and regular hidden Markov model (HMM) classifiers to identify GM12878 lymphoblastoid cell line specific regulatory regions using only the DNA sequence information from the human genome. The HMM was trained based on the assumption that the DNA sequence of a chromosome was generated using a Markovian process and the different regions in a chromosome such as regulatory regions, non-functional regions, etc were generated while the underlying process was in different states. On the other hand, the SVM classifier was trained to solve a binary classification problem using the k-spectrum string kernel [101]. Next, to use sequence evolution information along with sequence readout information, we selected the phylogenetic hidden Markov model (phylo-HMM) classifier. A phylo-HMM can be seen as a generator of multiple sequence alignments (MSA). It does so by running two Markov processes, one running along the branches of a phylogenetic tree and the other running along the length of a DNA sequence. We used MSA that not only included sequences from present day mammals, but also contained ancestral sequences that were constructed using the Ancestors 1.0 program [79]. This helped us capture better information about sequence evolution. In our knowledge this was the first attempt to use information from computationally inferred ancestral sequences in detecting regulatory regions.

We ran regular HMMs and phylo-HMMs with different structures and compared their results against each other to settle down on the best structure and variant of the HMM, but ended up observing that choice of the classifier depended on the desired false positive

rate. However, in the process, we visually analyzed a good number of predictions and discovered that the regular and the phylo-HMMs were making false positive predictions of different nature due to the different information they were being guided through. The false positive predictions made by the regular HMM were due to the lack of sequence evolution information, and the phylo-HMM was sometimes misguided by strong sequence conservation signal. We dealt with the problem of such false positive predictions by taking an intersection of the predictions made by the regular and the phylo-HMMs. This also helped us to settle for a single HMM based classifier that outperformed every other HMM classifier irrespective of the desired false positive rate. We further improved the performance of this classifier by combining its results with that of the SVM classifier. Thus, in the end, we had a classifier that was an ensemble of all three classifiers, combining their individual strengths.

The highly discriminating sequence patterns recognized by our classifiers were compared against the Homo sapiens Comprehensive Model Collection v11 core (HOCOMOCO v11 CORE) database [105] of known transcription factor binding motifs and a total of 91 matches were found.

5.2 Future Work

In this thesis, we trained machine learning models to identify cell type specific regulatory regions based on sequence data of concerned and related species. We found that the classifier obtained by taking an ensemble of the parallel six two-state regular HMM, parallel six two-state phylogenetic HMM, and the SVM trained on sequence data using the k-spectrum kernel gave us the best results within the acceptable false positive rate range. We see a large scope for both extending our current models to build better models, and for

coming up with other techniques to tackle the problem. The HMMs used in this work have the state durations of the underlying Markov process intrinsically modeled as geometric distributions, and it would be interesting to model these state durations as Poisson or Gaussian distributions as in the case of hidden semi-Markov models [108]; however, given its large computational time requirement kept us away from trying it. Even though fast implementations exist, they are not fast enough yet to be applicable to our problem setting. One more area to work on is the dependence of overlapping 6-mers despite knowing the underlying state. We did come up with three heuristics and compared their performance against each other; however, other tricks shall also be tried in the hunt for a better solution for the dependence of overlapping 6-mers. The sequence labelling task of identifying cell type specific regulatory regions should also be approached using other machine learning techniques, particularly deep learning techniques such as recurrent neural networks shall be a good fit as they have shown to be a success in the sequence labelling tasks for natural language processing.

Appendix A

Matches for Over-represented K-mers Recognized by Two-state Parallel Six Regular Hidden Markov Model

Below are the top fifty over-represented 6-mers recognized by the parallel six two-state regular hidden Markov model along with the motif ids of the transcription factor binding motifs from the Homo sapiens Comprehensive Model Collection v11 core (HOCOMOCO v11 CORE) database [105]. We mention motif ids instead of just transcription factor names to facilitate easy access to sequence signature of the motifs from the HOCOMOCO v11 CORE database.

CGCGCG - E2F5_HUMAN.H11MO.0.B, ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B

GCGGCG - TAF1_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TYY1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C

CGCCGC - TAF1_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TYY1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C

GCCGCG - ZFX_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B, MYCN_HUMAN.H11MO.0.A, TYY1_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A, KLF1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, KLF3_HUMAN.H11MO.0.B

CGCGGC - ZFX_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B, TYY1_HUMAN.H11MO.0.A, MYCN_HUMAN.H11MO.0.A,

MYC_HUMAN.H11MO.0.A, KLF1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, KLF3_HUMAN.H11MO.0.B

CGGCGG - TAF1_HUMAN.H11MO.0.A, MBD2_HUMAN.H11MO.0.B, TYY1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, MXI1_HUMAN.H11MO.0.A

CCGCCG - TAF1_HUMAN.H11MO.0.A, MBD2_HUMAN.H11MO.0.B, TYY1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, MXI1_HUMAN.H11MO.0.A

CCGCGG - AP2B_HUMAN.H11MO.0.B, NR1H4_HUMAN.H11MO.0.B

CCCGCG - AP2B_HUMAN.H11MO.0.B, E2F4_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, NRF1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, E2F7_HUMAN.H11MO.0.B, NR1H4_HUMAN.H11MO.0.B

CGGCGC - CTCFL_HUMAN.H11MO.0.A, ZN335_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B, KLF12_HUMAN.H11MO.0.C

CGCGGG - AP2B_HUMAN.H11MO.0.B, E2F4_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, NRF1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, E2F7_HUMAN.H11MO.0.B, NR1H4_HUMAN.H11MO.0.B

GCGCCG - CTCFL_HUMAN.H11MO.0.A, ZN335_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B, KLF12_HUMAN.H11MO.0.C

GCGCGC - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, AHR_HUMAN.H11MO.0.B

CCGGCG - MBD2_HUMAN.H11MO.0.B, ETV1_HUMAN.H11MO.0.A, SP2_HUMAN.H11MO.0.A, MECP2_HUMAN.H11MO.0.C

CGCCGG - MBD2_HUMAN.H11MO.0.B, ETV1_HUMAN.H11MO.0.A, SP2_HUMAN.H11MO.0.A, MECP2_HUMAN.H11MO.0.C

CGCGGA - HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, ELF1_HUMAN.H11MO.0.A, MECP2_HUMAN.H11MO.0.C

TCCGCG - HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, ELF1_HUMAN.H11MO.0.A, MECP2_HUMAN.H11MO.0.C

CGCGCT - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, MYC_HUMAN.H11MO.0.A, MAX_HUMAN.H11MO.0.A

AGCGCG - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, MYC_HUMAN.H11MO.0.A, MAX_HUMAN.H11MO.0.A

CGCGAC - AHR_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B, TFE3_HUMAN.H11MO.0.B, TFEB_HUMAN.H11MO.0.C, MITF_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A

CGCGAG - KAISO_HUMAN.H11MO.0.A, HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B

CCGCGA - HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, KAISO_HUMAN.H11MO.0.A, TYY1_HUMAN.H11MO.0.A

CTCGCG - KAISO_HUMAN.H11MO.0.A, HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B

TCGGCG - ZN335_HUMAN.H11MO.0.A

CGCCGA - ZN335_HUMAN.H11MO.0.A

CGGCGA - TAF1_HUMAN.H11MO.0.A, MBD2_HUMAN.H11MO.0.B

TCGCGG - HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, KAISO_HUMAN.H11MO.0.A, TYY1_HUMAN.H11MO.0.A

GTCGCG - AHR_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B, TFE3_HUMAN.H11MO.0.B, TFEB_HUMAN.H11MO.0.C, MITF_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A

TCGCCG - TAF1_HUMAN.H11MO.0.A, MBD2_HUMAN.H11MO.0.B

TGCGCG - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B

CGAGCG - ZBT14_HUMAN.H11MO.0.C, PAX5_HUMAN.H11MO.0.A

CGCGCA - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B

CGCTCG - ZBT14_HUMAN.H11MO.0.C, PAX5_HUMAN.H11MO.0.A

CGGCCG - MBD2_HUMAN.H11MO.0.B, HINFP_HUMAN.H11MO.0.C

GCGCGG - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A,

E2F1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A

CGCGTC - E2F2_HUMAN.H11MO.0.B, ATF6A_HUMAN.H11MO.0.B, HIF1A_HUMAN.H11MO.0.C

CCGCGC - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A

CGTCGC - RFX1_HUMAN.H11MO.0.B, MYCN_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A

GCGACG - RFX1_HUMAN.H11MO.0.B, MYCN_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A

GACGCG - E2F2_HUMAN.H11MO.0.B, ATF6A_HUMAN.H11MO.0.B, HIF1A_HUMAN.H11MO.0.C

CGACCG - ZBT48_HUMAN.H11MO.0.C

GGCGGC - TYY1_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TAF1_HUMAN.H11MO.0.A, ZFP42_HUMAN.H11MO.0.A, CTCFL_HUMAN.H11MO.0.A, MXI1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, INSM1_HUMAN.H11MO.0.C, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, CTCF_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A

GCCGCC - TYY1_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TAF1_HUMAN.H11MO.0.A, ZFP42_HUMAN.H11MO.0.A, CTCFL_HUMAN.H11MO.0.A, MXI1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, INSM1_HUMAN.H11MO.0.C, KLF12_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, CTCF_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A

CGGTCG - ZBT48_HUMAN.H11MO.0.C

GGCCGC - ZFX_HUMAN.H11MO.0.A, ZFP42_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TYY1_HUMAN.H11MO.0.A, HEN1_HUMAN.H11MO.0.C, MBD2_HUMAN.H11MO.0.B, KLF1_HUMAN.H11MO.0.A, CTCF_HUMAN.H11MO.0.A, KLF3_HUMAN.H11MO.0.B, HINFP_HUMAN.H11MO.0.C

GCGGCC - ZFX_HUMAN.H11MO.0.A, ZFP42_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, TYY1_HUMAN.H11MO.0.A, HEN1_HUMAN.H11MO.0.C, MBD2_HUMAN.H11MO.0.B, KLF1_HUMAN.H11MO.0.A, CTCF_HUMAN.H11MO.0.A, KLF3_HUMAN.H11MO.0.B, HINFP_HUMAN.H11MO.0.C

TTCGCG - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, TFDP1_HUMAN.H11MO.0.C, E2F3_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A

GGGGCG - SP1_HUMAN.H11MO.0.A, SP3_HUMAN.H11MO.0.B, E2F4_HUMAN.H11MO.0.A, KLF9_HUMAN.H11MO.0.C, SP4_HUMAN.H11MO.0.A, E2F6_HUMAN.H11MO.0.A, INSM1_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, CTCF_HUMAN.H11MO.0.A, HEN1_HUMAN.H11MO.0.C, EGR1_HUMAN.H11MO.0.A, KLF6_HUMAN.H11MO.0.A, KLF8_HUMAN.H11MO.0.C, ZBT14_HUMAN.H11MO.0.C, ZBT17_HUMAN.H11MO.0.A, KLF12_HUMAN.H11MO.0.C, CTCFL_HUMAN.H11MO.0.A, SP2_HUMAN.H11MO.0.A, E2F7_HUMAN.H11MO.0.B, E2F1_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, KLF1_HUMAN.H11MO.0.A, PATZ1_HUMAN.H11MO.0.C, KLF3_HUMAN.H11MO.0.B, ZBT7A_HUMAN.H11MO.0.A, TAL1_HUMAN.H11MO.0.A, KLF15_HUMAN.H11MO.0.A, ZN281_HUMAN.H11MO.0.A, SRBP2_HUMAN.H11MO.0.B,

Appendix B

Matches for Over-represented K-mers Recognized by Two-state Parallel Six Phylogenetic Hidden Markov Model

Below are the top fifty over-represented 6-mers recognized by the parallel six two-state phylogenetic hidden Markov model along with the motif ids of the transcription factor binding motifs from the Homo sapiens Comprehensive Model Collection v11 core (HOCOMOCO v11 CORE) database [105].

TCGCGA - KAISO_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B

TACGCG - EPAS1_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B

CGTACG - ARNT_HUMAN.H11MO.0.B, EPAS1_HUMAN.H11MO.0.B

CGCGTA - EPAS1_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B

CGCGAA - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, TFDP1_HUMAN.H11MO.0.C, E2F3_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A

TCGCGT - AHR_HUMAN.H11MO.0.B, ATF3_HUMAN.H11MO.0.A, TFEB_HUMAN.H11MO.0.C, TFE3_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B, MITF_HUMAN.H11MO.0.A, BHE40_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A

CGCGTA - EPAS1_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B

CGCGAA - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, TFDP1_HUMAN.H11MO.0.C, E2F3_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A

TTCGCG - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, TFDP1_HUMAN.H11MO.0.C, E2F3_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A

CGCGAT - KAISO_HUMAN.H11MO.0.A, ZBT14_HUMAN.H11MO.0.C, TYY1_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B

TCGCGA - KAISO_HUMAN.H11MO.0.A, E2F2_HUMAN.H11MO.0.B

CGCGTT - ZBT14_HUMAN.H11MO.0.C, ELF1_HUMAN.H11MO.0.A

CGAACG - HINFP_HUMAN.H11MO.0.C

CGTTCG - HINFP_HUMAN.H11MO.0.C

CGCGAA - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, TFDP1_HUMAN.H11MO.0.C, E2F3_HUMAN.H11MO.0.A, E2F1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A

TGCGCA - NRF1_HUMAN.H11MO.0.A, CEBPE_HUMAN.H11MO.0.A, CEBPD_HUMAN.H11MO.0.C, MTF1_HUMAN.H11MO.0.C, CEBPB_HUMAN.H11MO.0.A

GCGCAT - NRF1_HUMAN.H11MO.0.A

TGCGCA - NRF1_HUMAN.H11MO.0.A, CEBPE_HUMAN.H11MO.0.A, CEBPD_HUMAN.H11MO.0.C, MTF1_HUMAN.H11MO.0.C, CEBPB_HUMAN.H11MO.0.A

CACGCG - AHR_HUMAN.H11MO.0.B, MAX_HUMAN.H11MO.0.A, TFEB_HUMAN.H11MO.0.C, TFE3_HUMAN.H11MO.0.B, BHE40_HUMAN.H11MO.0.A, MYCN_HUMAN.H11MO.0.A, MITF_HUMAN.H11MO.0.A, BMAL1_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A, CLOCK_HUMAN.H11MO.0.C, MXI1_HUMAN.H11MO.0.A

CGCGTG - AHR_HUMAN.H11MO.0.B, MAX_HUMAN.H11MO.0.A, TFEB_HUMAN.H11MO.0.C, TFE3_HUMAN.H11MO.0.B, BHE40_HUMAN.H11MO.0.A, MYCN_HUMAN.H11MO.0.A, MITF_HUMAN.H11MO.0.A, BMAL1_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A, CLOCK_HUMAN.H11MO.0.C, MXI1_HUMAN.H11MO.0.A

TGGCGA - RFX1_HUMAN.H11MO.0.B, RFX2_HUMAN.H11MO.0.A, TYY1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A

GCGCGT - ZBT14_HUMAN.H11MO.0.C, HIF1A_HUMAN.H11MO.0.C, MAX_HUMAN.H11MO.0.A, BHE40_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A, ARNT_HUMAN.H11MO.0.B, EPAS1_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B

CGCATG - NRF1_HUMAN.H11MO.0.A, PAX6_HUMAN.H11MO.0.C

CGCGCA - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B

CGCCAT - TYY1_HUMAN.H11MO.0.A, THAP1_HUMAN.H11MO.0.C, ZFP42_HUMAN.H11MO.0.A, TAF1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, RFX1_HUMAN.H11MO.0.B, RFX2_HUMAN.H11MO.0.A, SOX17_HUMAN.H11MO.0.C, HXB4_HUMAN.H11MO.0.B

CATGCG - NRF1_HUMAN.H11MO.0.A, PAX6_HUMAN.H11MO.0.C

TGCGCG - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B

CGCGCA - ZBT14_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B

CGCGAG - KAISO_HUMAN.H11MO.0.A, HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B

CGTGCG - MYCN_HUMAN.H11MO.0.A, MTF1_HUMAN.H11MO.0.C, HIF1A_HUMAN.H11MO.0.C, AHR_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, HINFP_HUMAN.H11MO.0.C

GCGCAA - CEBPE_HUMAN.H11MO.0.A, CEBPD_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, CEBPB_HUMAN.H11MO.0.A, MTF1_HUMAN.H11MO.0.C, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B, HIC1_HUMAN.H11MO.0.C, ZN449_HUMAN.H11MO.0.C, CEBPA_HUMAN.H11MO.0.A, E2F5_HUMAN.H11MO.0.B

TTGCGC - CEBPE_HUMAN.H11MO.0.A, CEBPD_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, CEBPB_HUMAN.H11MO.0.A, MTF1_HUMAN.H11MO.0.C, NRF1_HUMAN.H11MO.0.A, AHR_HUMAN.H11MO.0.B, HIC1_HUMAN.H11MO.0.C, ZN449_HUMAN.H11MO.0.C, CEBPA_HUMAN.H11MO.0.A, E2F5_HUMAN.H11MO.0.B

CGCACG - MYCN_HUMAN.H11MO.0.A, MTF1_HUMAN.H11MO.0.C, HIF1A_HUMAN.H11MO.0.C, AHR_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, HINFP_HUMAN.H11MO.0.C

TTGCGA - CEBPE_HUMAN.H11MO.0.A, IKZF1_HUMAN.H11MO.0.C, AHR_HUMAN.H11MO.0.B, KAISO_HUMAN.H11MO.0.A

CCGCGA - HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B, KAISO_HUMAN.H11MO.0.A, TYY1_HUMAN.H11MO.0.A

GCGTAC - AHR_HUMAN.H11MO.0.B, ARNT_HUMAN.H11MO.0.B, EPAS1_HUMAN.H11MO.0.B

CGCGGT - ZBT14_HUMAN.H11MO.0.C, NR1H4_HUMAN.H11MO.0.B, RFX1_HUMAN.H11MO.0.B

CTCGCG - KAISO_HUMAN.H11MO.0.A, HINFP_HUMAN.H11MO.0.C, E2F2_HUMAN.H11MO.0.B

GTCGCG - AHR_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B, TFE3_HUMAN.H11MO.0.B, TFEB_HUMAN.H11MO.0.C, MITF_HUMAN.H11MO.0.A, USF1_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A

CTAGCG - HINFP_HUMAN.H11MO.0.C, KAISO_HUMAN.H11MO.0.A

CGCGTC - E2F2_HUMAN.H11MO.0.B, ATF6A_HUMAN.H11MO.0.B, HIF1A_HUMAN.H11MO.0.C

CGCTAG - HINFP_HUMAN.H11MO.0.C, KAISO_HUMAN.H11MO.0.A

TTTCGC - E2F2_HUMAN.H11MO.0.B, IRF9_HUMAN.H11MO.0.C, E2F5_HUMAN.H11MO.0.B, IRF7_HUMAN.H11MO.0.C, CEBPE_HUMAN.H11MO.0.A, NFKB1_HUMAN.H11MO.0.A, IRF2_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, KAISO_HUMAN.H11MO.0.A

CCGCGT - MAX_HUMAN.H11MO.0.A, MYCN_HUMAN.H11MO.0.A, MYC_HUMAN.H11MO.0.A, ELF1_HUMAN.H11MO.0.A, ATF6A_HUMAN.H11MO.0.B, MXI1_HUMAN.H11MO.0.A, BMAL1_HUMAN.H11MO.0.A

GCGCGA - E2F2_HUMAN.H11MO.0.B, E2F5_HUMAN.H11MO.0.B, ZBT14_HUMAN.H11MO.0.C, E2F1_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A, TFDP1_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A, E2F4_HUMAN.H11MO.0.A, KAISO_HUMAN.H11MO.0.A, E2F7_HUMAN.H11MO.0.B

TACGCA - HLF_HUMAN.H11MO.0.C, AHR_HUMAN.H11MO.0.B, CEBPE_HUMAN.H11MO.0.A, NRF1_HUMAN.H11MO.0.A, ZN260_HUMAN.H11MO.0.C

TTCCGC - PAX6_HUMAN.H11MO.0.C, ZN140_HUMAN.H11MO.0.C, ELF2_HUMAN.H11MO.0.C, GABPA_HUMAN.H11MO.0.A,

ETV1_HUMAN.H11MO.0.A, E2F7_HUMAN.H11MO.0.B,
ELK1_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B,
ELK4_HUMAN.H11MO.0.A, ELF1_HUMAN.H11MO.0.A,
E2F4_HUMAN.H11MO.0.A, E2F3_HUMAN.H11MO.0.A,
TFDP1_HUMAN.H11MO.0.C, E2F6_HUMAN.H11MO.0.A,
E2F1_HUMAN.H11MO.0.A, OSR2_HUMAN.H11MO.0.C,
ELF5_HUMAN.H11MO.0.A, HINFP_HUMAN.H11MO.0.C

CGCAAC - AHR_HUMAN.H11MO.0.B, E2F2_HUMAN.H11MO.0.B,
CEBPD_HUMAN.H11MO.0.C, HIC1_HUMAN.H11MO.0.C,
ZN449_HUMAN.H11MO.0.C

References

[1] M. Zhang. Statistical features of human exons and their flanking regions. *Human Molecular Genetics*, 7(5):919–932, Jan 1998.

[2] File:eukaryotic transcription.svg. https://commons.wikimedia.org/wiki/File:Eukaryotic Transcription.svg.

[3] S. Seguir. Structure of gene. https://pt.slideshare.net/Sayali28/fine-structure-of-gene-57949681/14.

[4] Gene family: Cyclin dependent kinases (CDK). https://www.genenames.org/cgi-bin/genefamilies/set/496.

[5] Lifespan of human body cells. https://www.medicalsciencenavigator.com/physiology-of-self-renewal/.

[6] B. Cornell. Gene expression — BioNinja. http://ib.bioninja.com.au/higher-level/topic-7-nucleic-acids/72-transcription-and-gene/gene-expression.html.

[7] Swiss Institute of Bioinformatics European Bioinformatics Institute Protein Information Resource. Protein S100-A9, May 2018. http://www.uniprot.org/uniprot/P06702.

[8] T. Kwok. Severe acute respiratory syndrome in elderly patients. *Severe Acute Respiratory Syndrome*, pages 159–164, 2004.

[9] OpenStax College. *Concepts of Biology*. OpenStax CNX. http://cnx.org/contents/b3c1e1d2-839c-42b0-a314-e119a8aafbdd@9.25.

[10] J. M. Vaquerizas, S. K. Kummerfeld, S. A. Teichmann, and N. M. Luscombe. A census of human transcription factors: function, expression and evolution. *Nature Reviews Genetics*, 10(4):252–263, 2009.

[11] L. A. Pennacchio. Enhancers: Five essential questions. www.nature.com/articles/nrg3458.

[12] R. Staden. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Research*, 12(1 Part 2):505–519, 1984.

[13] C. Chai, Z. Xie, and E. Grotewol. SELEX (Systematic Evolution of Ligands by EXponential Enrichment), as a powerful tool for deciphering the protein-DNA interaction space. *Methods in Molecular Biology Plant Transcription Factors*, pages 249–258, 2011.

[14] Transcription factor.
https://en.wikipedia.org/wiki/Transcription_factor.

[15] The structure of a transcription factor.
https://study.com/cimages/multimages/16/TF_1.png.

[16] E. M. Blackwood. Going the distance: A current view of enhancer action. *Science*, 281(5373):60–63, Mar 1998.

[17] W. De Laat and D. Duboule. Topology of mammalian developmental enhancers and their regulatory landscapes. *Nature*, 502(7472):499–506, 2013.

[18] S. Kadauke and G. A. Blobel. Chromatin loops in gene regulation. *Biochimica et Biophysica Acta (BBA) - Gene Regulatory Mechanisms*, 1789(1):17–25, 2009.

[19] Control sites and the initiation of transcription. http://ebooks.dynamic-learning.co.uk/prod_content/extracted_books/9781471840470/OEBPS/images/374-1.jpg.

[20] J. Eeckhoute, R. Metivier, and G. Salbert. Defining specificity of transcription factor regulatory activities. *Journal of Cell Science*, 122(22):4027–4034, Dec 2009.

[21] Nucleosome.
https://www.nature.com/scitable/definition/nucleosome-nucleosomes-30.

[22] Pioneer factor.
https://en.wikipedia.org/wiki/Pioneer_factor.

[23] M. J. Van Baren, B. C. Koebbe, and M. R. Brent. Using N-SCAN or TWINSCAN to predict gene structures in genomic DNA sequences. *Current Protocols in Bioinformatics*, 2007.

[24] D. N. Arnosti and M. M. Kulkarni. Transcriptional enhancers: Intelligent enhanceosomes or flexible billboards? *Journal of Cellular Biochemistry*, 94(5):890–898, 2005.

[25] L. Narlikar and I. Ovcharenko. Identifying regulatory elements in eukaryotic genomes. *Briefings in Functional Genomics and Proteomics*, 8(4):215230, Apr 2009.

[26] D. Boffelli, M. A. Nobrega, and E. M. Rubin. Comparative genomics at the vertebrate extremes. *Nature Reviews Genetics*, 5(6):456–465, 2004.

[27] Beta thalassemia - genetics home reference. https://ghr.nlm.nih.gov/condition/beta-thalassemia.

[28] S. E. Antonarakis, S. H. Irkin, T. C. Cheng, A. F. Scott, J. P. Sexton, S. P. Trusko, S. Charache, and H. H. Kazazian. beta-Thalassemia in American Blacks: novel mutations in the "TATA" box and an acceptor splice site. *Proceedings of the National Academy of Sciences*, 81(4):1154–1158, Jan 1984.

[29] Pyruvate kinase deficiency. https://rarediseases.info.nih.gov/diseases/7514/pyruvate-kinase-deficiency.

[30] Z. Tong, Z. Yang, S. Patel, H. Chen, D. Gibbs, X. Yang, V. S. Hau, Y. Kaminoh, J. Harmon, E. Pearson, and et al. Promoter polymorphism of the erythropoietin gene in severe diabetic eye and kidney complications. *Proceedings of the National Academy of Sciences*, 105(19):6998–7003, May 2008.

[31] M. J. Solomon, P. L. Larsen, and A. Varshavsky. Mapping protein-DNA interactions in vivo with formaldehyde: Evidence that histone H4 is retained on a highly transcribed gene. *Cell*, 53(6):937–947, 1988.

[32] B. Ren. Genome-wide location and function of DNA binding proteins. *Science*, 290(5500):2306–2309, 2000.

[33] D. S. Johnson, A. Mortazavi, R. M. Myers, and B. Wold. Genome-wide mapping of in vivo protein-DNA interactions. *Science*, 316(5830):1497–1502, Aug 2007.

[34] C. Wei, Q. Wu, V. B. Vega, K. P. Chiu, P. Ng, T. Zhang, A. Shahab, H. C. Yong, Y. Fu, Z. Weng, and et al. A global map of p53 transcription-factor binding sites in the human genome. *Cell*, 124(1):207–219, 2006.

[35] A. P. Boyle, S. Davis, H. P. Shulha, P. Meltzer, E. H. Margulies, Z. Weng, T. S. Furey, and G. E. Crawford. High-resolution mapping and characterization of open chromatin across the genome. *Cell*, 132(2):311–322, 2008.

[36] J. R. Hesselberth, X. Chen, Z. Zhang, P. J. Sabo, R. Sandstrom, A. P. Reynolds, R. E. Thurman, S. Neph, M. S. Kuehn, W. S. Noble, and et al. Global mapping of protein-DNA interactions in vivo by digital genomic footprinting. *Nature Methods*, 6(4):283–289, 2009.

[37] O. Johansson, W. Alkema, W. W. Wasserman, and J. Lagergren. Identification of functional clusters of transcription factor binding motifs in genome sequences: the mscan algorithm. *Bioinformatics*, 19(Suppl 1):i169–i176, Mar 2003.

[38] A. P. Lifanov. Homotypic regulatory clusters in drosophila. *Genome Research*, 13(4):579–588, Jan 2003.

[39] M. Markstein, P. Markstein, V. Markstein, and M. S. Levine. Genome-wide analysis of clustered dorsal binding sites identifies putative target genes in the drosophila embryo. *Proceedings of the National Academy of Sciences*, 99(2):763–768, 2001.

[40] A. Wagner. Genes regulated cooperatively by one or more transcription factors and their identification in whole eukaryotic genomes. *Bioinformatics*, 15(10):776–784, Jan 1999.

[41] M. Blanchette, A. Bataille, and X. Chen. Genome-wide computational prediction of transcriptional regulatory modules reveals new insights into human gene expression. *Genome Research*, 16(5):656–668, Jan 2006.

[42] V. Matys. TRANSFAC(R): transcriptional regulation, from patterns to profiles. *Nucleic Acids Research*, 31(1):374–378, Jan 2003.

[43] W. W. Wasserman and J. W. Fickett. Identification of regulatory regions which confer muscle-specific gene expression. *Journal of Molecular Biology*, 278(1):167–181, 1998.

[44] W. Krivan and W. Wasserman. A predictive model for regulatory sequences directing liver-specific transcription. *Genome Research*, 11(9):1559–1566, 2001.

[45] M. C. Frith, U. Hansen, and Z. Weng. Detection of cis -element clusters in higher eukaryotic DNA. *Bioinformatics*, 17(10):878–889, Jan 2001.

[46] M. C. Frith, M. C. Li, and Z. Weng. Cluster-Buster: finding dense clusters of motifs in DNA sequences. *Nucleic Acids Research*, 31(13):3666–3668, Jan 2003.

[47] S. Sinha, E. Van Nimwegen, and E. D. Siggia. A probabilistic method to detect regulatory modules. *Bioinformatics*, 19(Suppl 1):i292–i301, Mar 2003.

[48] M. Brudno, C Do, G Cooper, M Kim, E Davydov, E Green, A Sidow, and S Batzoglu. LAGAN and Multi-LAGAN: Efficient tools for large-scale multiple alignment of genomic DNA. *Genome Research*, 13(4):721–731, Dec 2003.

[49] B. Morgenstern, K. Frech, A. Dress, and T. Werner. DIALIGN: finding local similarities by multiple sequence alignment. *Bioinformatics*, 14(3):290–294, Jan 1998.

[50] S. Sinha and X. He. MORPH: Probabilistic alignment combined with hidden markov models of cis-regulatory modules. *PLoS Computational Biology*, 3(11), 2007.

[51] I. Holmes and R. Durbin. Dynamic programming alignment accuracy. *Proceedings of the second annual international conference on Computational molecular biology - RECOMB 98*, 1998.

[52] O. Hallikas, K. Palin, N. Sinjushina, R. Rautiainen, J. Partanen, E. Ukkonen, and J. Taipale. Genome-wide prediction of mammalian enhancers based on analysis of transcription-factor binding affinity. *Cell*, 124(1):47–59, 2006.

[53] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[54] L. Elnitski, R. Hardison, J. Li, S. Yang, D. Kolbe, P. Eswara, M. O'Connor, S. Schwartz, W. Miller, F. Chiaromonte, and et al. Distinguishing regulatory DNA from neutral sites. *Genome Research*, 13(1):64–72, Jan 2003.

[55] D. Kolbe, J. Taylor, L. Elnitski, P. Eswara, J. Li, W. Miller, R. Hardison, and F. Chiaromonte. Regulatory potential scores from genome-wide three-way alignments of human, mouse, and rat. *Genome Research*, 14(4):700–707, Jan 2004.

[56] Q. Zhou and W. H. Wong. CisModule: De novo discovery of cis-regulatory modules by hierarchical mixture modeling. *Proceedings of the National Academy of Sciences*, 101(33):12114–12119, May 2004.

[57] B. Chan and D. Kibler. Using hexamers to predict cis-re gulatory motifs in drosophila. *BMC Bioinformatics*, Oct 2005.

[58] C. Fletez-Brant, D. Lee, A. S. Mccallion, and M. A. Beer. kmer-svm: a web server for identifying predictive regulatory sequence features in genomic data sets. *Nucleic Acids Research*, 41(W1), 2013.

[59] L. Narlikar, N. J. Sakabe, A. A. Blanski, F. E. Arimura, J. M. Westlund, M. A. Nobrega, and I. Ovcharenko. Genome-wide discovery of human heart enhancers. *Genome Research*, 20(3):381–392, 2010.

[60] A. Sandelin, W. Alkema, B. Lenhard, W. W. Wasserman, and P. Engstrom. JASPAR: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Research*, 32(90001), Jan 2004.

[61] L. Narlikar, R. Gord¢n, and A. J. Hartemink. A nucleosome-guided map of transcription factor binding sites in yeast. *PLoS Computational Biology*, 3(11):e215, 2007.

[62] G. D. Erwin, N. Oksenberg, R. M. Truty, D. Kostka, K. K. Murphy, N. Ahituv, K. S. Pollard, and J. A. Capra. Integrating diverse datasets improves developmental enhancer prediction. *PLoS Computational Biology*, 10(6):e1003677, 2014.

[63] C. Blatti, M. Kazemian, S. Wolfe, M. Brodsky, and S. Sinha. Integrating motif, DNA accessibility and gene expression data to build regulatory maps in an organism. *Nucleic Acids Research*, 43(8):3998–4012, 2015.

[64] Sequence labeling.
https://en.wikipedia.org/wiki/Sequence_labeling.

[65] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260269, 1967.

[66] L. E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In Oved Shisha, editor, *Inequalities III: Proceedings of the Third Symposium on Inequalities*, pages 1–8, University of California, Los Angeles, 1972. Academic Press.

[67] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[68] E. S. Krogh A. Mitchison Durbin, R. *Biological sequence analysis probabilistic models of proteins and nucleic acids*. Cambridge Univ. Press, 2013.

[69] W. J. Kent, C. W. Sugnet, T. S. Furey, K. M. Roskin, T. H. Pringle, A. M. Zahler, and A. D. Haussler. The human genome browser at UCSC. *Genome Research*, 12(6):996–1006, 2002.

[70] D. Karolchik, A. S. Hinrichs, T. S. Furey, K. M. Roskin, C. W. Sugnet, D. Haussler, and W. J. Kent. The UCSC Table Browser data retrieval tool. *Nucleic Acids Research*, 32(90001), Jan 2004.

[71] Multiple sequence alignment, May 2018. https://en.wikipedia.org/wiki/Multiple_sequence_alignment#cite_ref-feng1987progressive_8-0.

[72] J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.

[73] C. Magis, J. Taly, G. Bussotti, J. Chang, P. Di Tommaso, I. Erb, J. Espinosa-Carrasco, and C. Notredame. T-Coffee: Tree-Based Consistency Objective Function for Alignment Evaluation. *Methods in Molecular Biology Multiple Sequence Alignment Methods*, pages 117–129, 2013.

[74] R. C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, Aug 2004.

[75] M. Blanchette, W. J. Kent, C. Reimer, L. Elnitski, A. Smit, K. M. Roskin, R. Baertsch, and K. Rosenbloom. Aligning multiple genomic sequences with the threaded blockset aligner. *Genome Research*, 14(4):708–715, Jan 2004.

[76] S. Schwartz, W. J. Kent, A. Smit, Z. Zhang, R. Baertsch, R. C. Hardison, D. Haussler, and W Miller. Human-mouse alignments with BLASTZ. *Genome Research*, 13(1):103–107, Jan 2003.

[77] S. Schwartz, Z. Zhang, K.A. Frazer, A. Smit, C. Reimer, J. Bouck, R. Gibbs, R.C. Hardison, and W. Miller. PipMaker—a web server for aligning two genomic DNA sequences. *Genome Research*, 10(4):577–586, Jan 2000.

[78] R. S. Harris. *Improved pairwise alignment of genomic DNA. Ph.D. Thesis.* PhD thesis, 2007.

[79] A. B. Diallo, V. Makarenkov, and M. Blanchette. Ancestors 1.0: a web server for ancestral sequence reconstruction. *Bioinformatics*, 26(1):130–131, 2009.

[80] M. Blanchette, E. Green, W. Miller, and D. Haussler. Reconstructing large regions of an ancestral mammalian genome in silico. *Genome Research*, 14(12):2412–2423, 2004.

[81] Diallo A. Green E. Miller W. Haussler D. Blanchette, M. Computational reconstruction of ancestral DNA sequences. In *Methods in Molecular Biology: Phylogenomics*, chapter 11, pages 171–184. Springer, 2008.

[82] What is FASTA format?
https://zhanglab.ccmb.med.umich.edu/FASTA/.

[83] The Newick tree format.
http://evolution.genetics.washington.edu/phylip/newicktree.html.

[84] A. B. Diallo, V. Makarenkov, and M. Blanchette. Exact and heuristic algorithms for the indel maximum likelihood problem. *Journal of Computational Biology*, 14(4):446–461, 2007.

[85] J. Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981.

[86] J. Felsenstein and G. A. Churchill. A hidden Markov model approach to variation among sites in rate of evolution. *Molecular Biology and Evolution*, 13(1):93–104, Jan 1996.

[87] A. Siepel and D. Haussler. Combining phylogenetic and hidden Markov models in biosequence analysis. *Proceedings of the seventh annual international conference on Computational molecular biology - RECOMB 03*, 2003.

[88] Z. Yang. A spacetime process model for the evolution of DNA sequences. *Genetics*, 1995.

[89] C. M. Bishop. Sparse kernel machines. In *Pattern Recognition and Machine Learning*, chapter 7, pages 325–358. Springer New York, 2006.

[90] Genome reference consortium.
https://www.ncbi.nlm.nih.gov/grc.

[91] Displaying your own annotations in the genome browser.
http://genome.cse.ucsc.edu/goldenPath/help/customTrack.html.

[92] The ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414):57–74, 2012.

[93] Txn Factor ChIP Track Settings. http://genome.ucsc.edu/cgi-bin/hgTrackUi?db=hg19g=wgEncodeRegTfbsClusteredV3.

[94] B. J. Raney, M. S. Cline, K. R. Rosenbloom, T. R. Dreszer, K. Learned, G. P. Barber, L. R. Meyer, C. A. Sloan, V. S. Malladi, K. M. Roskin, and et al. ENCODE whole-genome data in the UCSC genome browser (2011 update). *Nucleic Acids Research*, 39(suppl_1), 2010.

[95] R. M. Kuhn, D. Haussler, and W. J. Kent. The UCSC genome browser and associated tools. *Briefings in Bioinformatics*, 14(2):144–161, 2012.

[96] ENCODE Project Common Cell Types. https://www.genome.gov/26524238/encode-project-common-cell-types/.

[97] Index of /goldenPath/hg38/multiz100way. http://hgdownload.soe.ucsc.edu/goldenPath/hg38/multiz100way/.

[98] Vertebrate Multiz alignment & conservation (100 species) track settings. http://genome.ucsc.edu/cgi-bin/hgTrackUi?db=hg38g=cons100way.

[99] Ostrich algorithm. https://en.wikipedia.org/wiki/Ostrich_algorithm.

[100] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98 Lecture Notes in Computer Science*, pages 137–142, 1998.

[101] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for svm protein classification. *Biocomputing 2002*, pages 564–575.

[102] DNase HS Track Settings. https://genome.ucsc.edu/cgi-bin/hgTrackUi?db=hg38g=wgEncodeRegDnase.

[103] ENCODE Regulation Super-track Settings. https://genome.ucsc.edu/cgi-bin/hgTrackUi?g=wgEncodeReg.

[104] S. Gupta, J. Stamatoyannopolous, T. Bailey, and W. S. Noble. Quantifying similarity between motifs. *Genome Biology*, page R24, 2007.

[105] I. V. Kulakovskiy, I. E. Vorontsov, I. S Yevshin, R. N. Sharipov, A. D. Fedorova, E. I. Rumynskiy, Y. A. Medvedeva, A. Magana-Mora, V. B. Bajic, D. A. Papatsenko, F. A. Kolpakov, and V. J. Makeev. HOCOMOCO: towards a complete collection of transcription factor binding models for human and mouse via large-scale ChIP-seq analysis. *Nucleic Acids Research*, 46(D1):D252–D259, 2018.

[106] European Bioinformatics Institute Protein Information Resource SIB Swiss Institute of Bioinformatics. ETS-related transcription factor ELF-1. https://www.uniprot.org/uniprot/P32519.

[107] European Bioinformatics Institute Protein Information Resource SIB Swiss Institute of Bioinformatics. Transcription factor E2F2. https://www.uniprot.org/uniprot/Q14209.

[108] S. Yu. Hidden semi-markov models. *Artificial Intelligence*, 174(2):215 – 243, 2010.