Classification of Regulatory Sequences in the Human Genome Using High-Order Generalized Hidden Markov Models

by
David Taub

Supervised by
Prof. Tommy Kaplan

THE HEBREW
UNIVERSITY
OF JERUSALEM

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

May 2020

The Rachel and Selim Benin School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel

Acknowledgments

My deepest gratitude goes to my supervisor, Professor Tommy Kaplan, who sparked my curiosity for the nature of DNA in one of his open lectures I attended as an undergraduate student. In our many talks since then, Professor Kaplan's insights in the field of computational biology increased my interest even more. He taught me that patience and methodological work are the irreplaceable properties of any good researcher.

So thank you for your guidance and ideas during my participation in this work, and for the positive and pleasant atmosphere you always maintain in your research group. Your support and understanding gave me confidence through the hard parts of this road, and kept me going.

I would also wish to thank my mother, Dr. Jenny Taub and my sister Matty Ariel for their careful review of this work. I am in debt for the many helpful suggestions they provided, which greatly improved the language and content of this thesis.

Abstract

Enhancers are regulatory DNA sequences that, when bound to proteins called transcription factors, increase the likelihood of transcription of the enhancer target genes. Regulation of transcription is an important form of control of gene expression, and the activity of enhancers plays a significant role in the stage-specific and tissue-specific regulation of genes. It has been shown over the years that genetic variations within enhancer sequences might cause cell behavior modifications and diseases. The structural rules and nuances of enhancers are not fully understood yet, though it has been shown that transcription factors tend to bind to them at unique and over-represented motifs called transcription factor binding sites. Enhancer activity can be detected by the epigenetic data from the local environment around its position. The main indicators for an enhancer lay in the adjacent histone modifications, around which the flanks of the enhancer are wrapped. The enhancer itself tends to be spatially accessible for biochemical interactions between the DNA and the proteins around it. Though useful, the epigenetic data are often noisy and require a costly extraction process of specific cells from a tissue sample, which is not necessarily practical for all cell types and their different stages. An alternative approach to enhancer detection is to observe the genetic content of its sequence, since it contains all the essential information for the DNA to act as an enhancer. Over the years, it was demonstrated *in vivo* that the cell requires no other mechanism than the genetic sequence in order to regulate its gene expression. With that in mind, this work presents a computational approach for the detection of enhancers based only on their sequences, and in an unsupervised manner. We created a higher-order positional weight matrix-based hidden Markov model (HOP-HMM), with two kinds of states: one which emits transcription factor binding sites by using a positional weight matrix model, and one which emits single nucleotides with high-order dependency on previously emitted nucleotides. Compared to a regular hidden Markov model, this model learns a more complex underlying structure of DNA sequences, containing both binding site motifs and high-order distribution of nucleotides in between them. We will first review the biological background of enhancers, specifically in humans. Then we will review in depth the background of Markov and hidden Markov models, and discuss how to calculate the likelihood of a sequence based on these models. We will describe our generalized model in detail and develop the expectation-maximization and Viterbi algorithms for hidden Markov models, followed by the adjustments needed for our generalized model. The implementation of these algorithms is demonstrated by applying them to a synthetic dataset of enhancer-like sequences created by using the generative properties of the generalized model. We will use the model in a controlled way in order to assess it by inferring estimated parameters and comparing them to the real parameters used to create the dataset. Finally, we will apply the expectation-maximization algorithm to train a HOP-HMM on human enhancer sequences, which were selected on the basis of the epigenetic data of the Roadmap project. We will demonstrate the capabilities of the model by comparing its estimations to the epigenetic tracks, showing that it can predict both the loci of enhancers and in which tissues they will be active, without any exposure to epigenetic data.

Contents

1 Introduction

1.1 Background

The genome of every living organism contains the inherited information which defines its complex structure and function. The genome is built out of two intertwined strands of deoxyribonucleic acid (DNA) molecules called nucleotides, that form that form a double helical structure. Nucleotides can be one of 4 different basic bases: cytosine, guanine, adenine, and thymine or in short A,C,G and T respectively. The nucleotides are organized in pairs called base pairs, with each of the paired nucleotides being complementary to the other and providing redundancy.

Proteins are macromolecules, which play various roles and functions within organisms. They have the structure of a polymer built out of 20 different amino acids, whose order and configuration are encoded in genes (genetic segments within the genome). Through transcription followed by translation processes, the genes are expressed and result in the formation of proteins. During the transcription process the gene is read and transcribed into a single strand sequence of ribonucleic acids (RNA). Next, the formed RNA strand, which at this stage is called messenger RNA (mRNA), is translated by a complex molecule called the ribosome. The mRNA sequence is built out of triplets of nucleotides called codons, which are read by the ribosome and instruct it how to generate the sequence of amino acids constituting the required protein.

Gene sequences are composed of fragmented coding regions (exons) and noncoding regions (introns). Following its formation, the RNA molecule undergoes splicing, during which the introns are removed from the strand, and only the exons mature into mRNA molecules which are then translated into proteins. Counter intuitively, even though the exons hold the instructions for proteins formation, the complexity of the organism is not correlated to the number of exons or their total length in its genome. For example, both humans and *Caenorhabditis elegans* roundworms have about 19,000 genes with roughly the same total exon length and number (Ainscough et al., 1998; Ezkurdia et al., 2014), despite the greater diversity and complexity of the former. Although the genes are responsible for the variety of proteins a cell is able to produce, the complexity of an organism (i.e. the number of different cell types specializing in different tasks) stems from the gene regulation mechanisms. In the case of humans, the genome is 3.23 Gb-long and it is estimated that the total length of gene-regulating regions involves 10-20% of it (Pennacchio et al., 2015), compared to only 1% for exon regions (Ng et al., 2009).

Enhancers are non-coding regulatory DNA sequences which play a key role in the regulation of gene expression. In humans, there are hundreds of thousands of enhancers scattered over the non-coding regions of the genome, usually of a length between 100-1000 base pairs (bp). When the enhancer is activated, the DNA folding draws it spatially closer to another type of regulatory element called promoter, resulting in the translation of the gene adjacent to the promoter (see figure 1). The gene expressed by this activation process, also known as the target gene of the enhancer, can be located up to one megabase pair (Mb) upstream or downstream from its activating enhancer, since enhancers generally function independently of orientation (Williamson et al., 2011). Moreover, the gene-enhancer connection is not exclusive, and it has been shown that the most common case is that each enhancer has several target genes and vice-versa (Fishilevich et al., 2017).

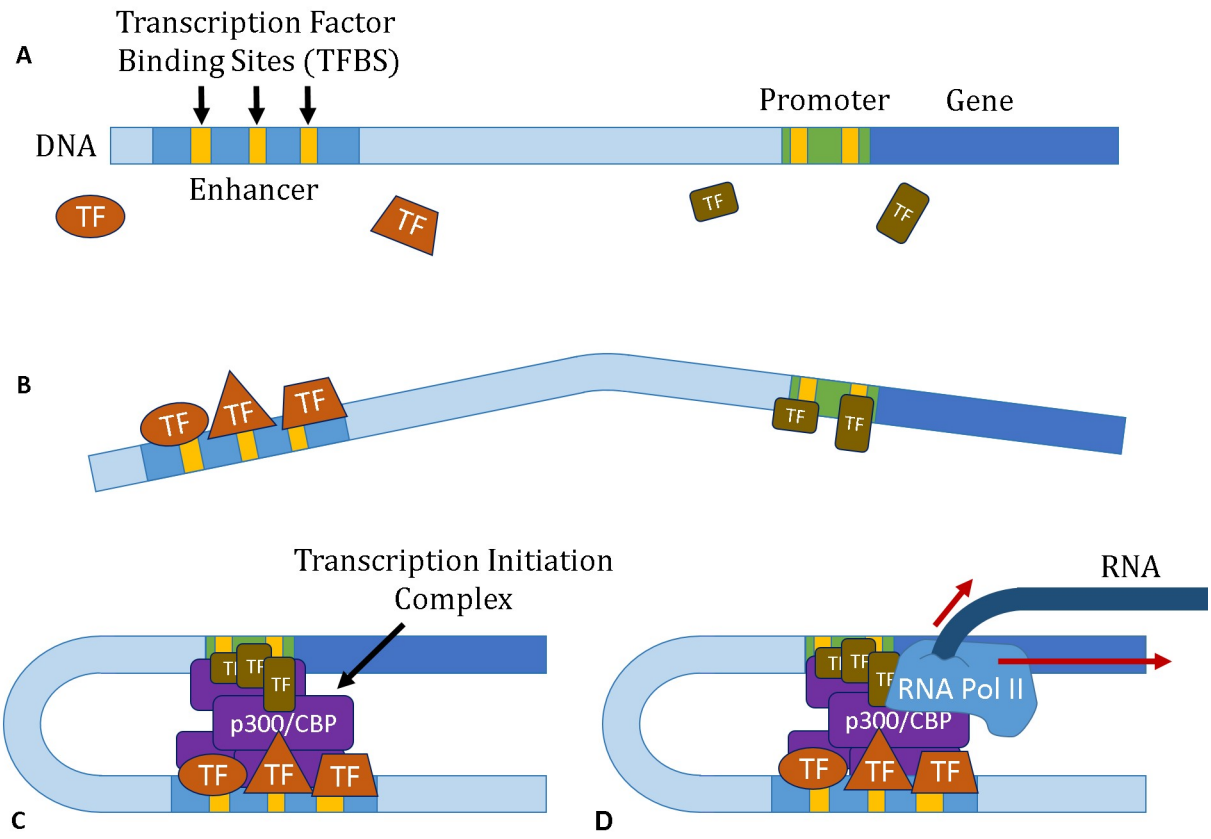


Figure 1: The stages of the activation process of an enhancer. **A)** A distal enhancer and upstream promoter and gene, positioned on a DNA strand. The sequences of both the enhancer and the promoter contain multiple TFBSs. **B)** The DNA folds and recruits TFs from its surrounding environment. **C)** Other cofactor proteins are recruited and form the transcription initiation complex, binding the two parts of the folded DNA strand to each other. **D)** The RNA Polymerase II is bound to the formed complex and starts moving along the gene, while generating a new RNA molecule.

An enhancer is described as being in an active status when it is causing the expression of its target gene, which does not occur evenly across different types of cells. The activity of the enhancer sequence plays a critical role in the resulting type of cells. In a large *in vivo* enhancer assay by Visel et al. (2007), fertilized mouse eggs were injected with enhancer sequences adjacent to a minimal promoter and a LacZ reporter gene, encoding an enzyme protein with a blue color. Since the injected DNA were synthesized, the enhancer, the promoter and the reporter gene bore no epigenetic information, and they were integrated into the mouse genome in an arbitrary position. The enhancer sequences in the injected DNA originated from the human and mouse genomes, and each enhancer was injected into a different mouse egg. When the transgenic embryos were photographed after 11.5 days some had a distinctive anatomical pattern, such as blue limbs or blue spine, depending on the injected DNA sequence. These results imply that for many DNA sequences, the DNA code possesses by itself the potential to become a specific tissue enhancer, despite the absence of epigenetic information.

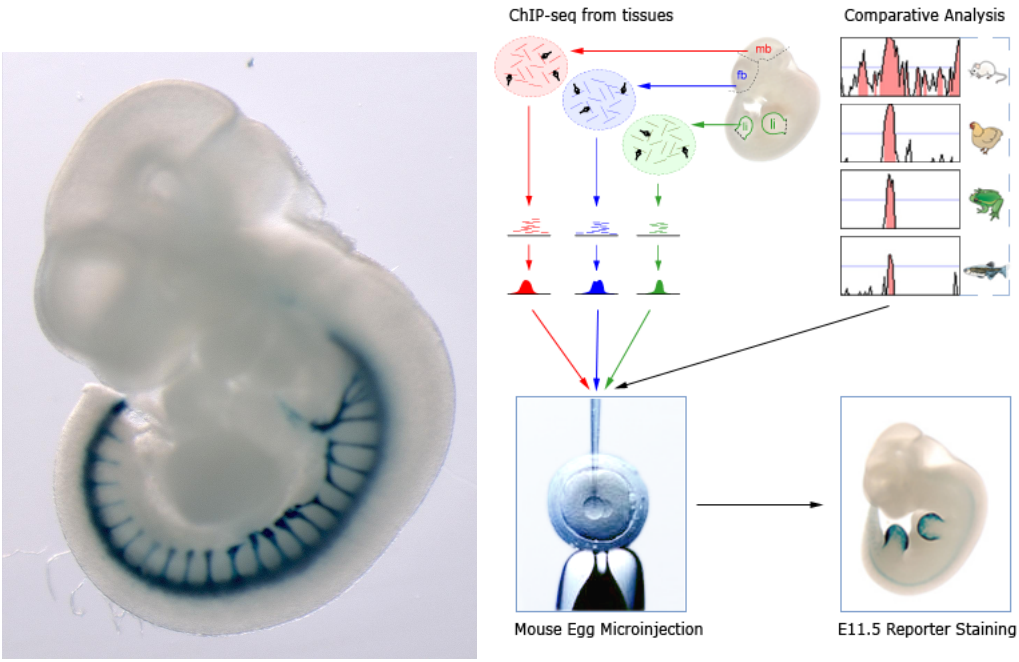


Figure 2: Transgenic mouse embryos on the 11.5 day with distinct anatomical pattern of enhancer activation in specific tissue types. In a large enhancer assay by Visel et al. (2007) described on the right, fertilized eggs were injected *in vivo* with synthetic DNA sequences which contained an enhancer, a minimal promoter and a LacZ reporter gene. On the left, a human enhancer known to be related to neurons in the dorsal roots of the spinal nerves became activated in these tissues and caused to the expression of the reporter gene that was coupled to it. Embryo 2 of experiment hs51, taken from the VISTA Enhancer Browser [`https://enhancer.lbl.gov/`].

Transcription factors (TF) are proteins that bind to the DNA, and together with other cofactor proteins initiate the gene transcription process of the DNA sequence. TFs tend to bind to their transcription factor binding sites (TFBS), which are motifs of nucleotides in the DNA sequence. The average length of TFBSs in humans is 12 bp (Kulakovskiy et al., 2011), and they are highly conserved between various species (Doniger et al., 2005). When analyzing a tissue sample for TF and cofactors interaction density with the DNA, the chromatin immunoprecipitation sequencing (ChIP-seq) method is used to probe the amount of proteins in affinity to the DNA strands. Briefly, this method involves applying antibodies on cross-linked and sheared DNA, and allowing these antibodies to bind to their target proteins. The bound antibodies and target proteins are precipitated together with the linked DNA strands which are then released, and undergo massive parallel sequencing. A ChIP-seq assay *in vivo* by Visel et al. (2009) has shown that ChIP-seq mapping of p300/CBP cofactors can accurately predict the location of enhancers. Genome-wide association studies (GWAS) of ChIP-seq found that different TFs have different and distinct distributions of TFBS (Khan et al., 2018).

The TFBSs in both enhancers and promoters are critical for their correct regulatory activity. Multiple studies have shown that genetic alterations in TFBSs inside enhancers can affect the expression of their target genes and are a major cause of various human diseases (Kreimer et al., 2017; Miguel-Escalada et al., 2015; Soldner et al., 2016; Smemo et al., 2012; Benko et al., 2009; Emison et al., 2005; Lettice et al., 2003). From the sequence aspect, enhancers and promoters have a similar structure: both have different nucleotide frequencies compared to other parts of the genome, and both contain TFBSs tiled inside background sequences.

Folding of the DNA allows enhancer-promoter interactions, in which the TFs play a major part. Once bounded to the DNA, the TFs recruit other protein cofactors, and together they form a transcription preinitiation complex (PIC) consisting of a very large assembly of proteins. Out of the tens of proteins

constructing the PIC, the sub-unit RNA polymerase (RNA pol II) has the important role of transcribing the adjacent gene. It slides along the double-stranded DNA and opens it until one strand of nucleotides is exposed and becomes a template for RNA synthesis.

Though it is tempting to imagine each TF as having a corresponding TFBS with a single motif of nucleotides that fits it, modeling the kinetic and thermodynamic aspects involved in the DNA-protein interaction is far from simple (Winter et al., 1981). Each sequence of nucleotides has the likelihood to form a bond, which is not simple to calculate analytically. In order to generate a simplistic yet statistically accurate model representing the TF binding potential of a DNA sequence, i.e. $P(x_{1:n}|binding)$, we need to assume an independence between positions, as well as a small range of influence of the sequence around the binding site. For samples of such distribution, the peaks of the ChIP-seq readings marking the TF binding are often used, from which a binding site "grammar" can be modeled. Position weight matrix (PWM), as introduced by Stormo et al. (1982), is the most commonly used probabilistic model to address this task. The underlying assumption of the PWM model is that the probability of every position at a TFBS is calculated independently, and therefore the total motif probability is a multiplication of its per-position probabilities:

$$P(x_{1:J}|binding) = \prod_{j\in[n]} P\left(x_j|binding\right)$$

Where J is the length of the relevant sequences affected by the binding event, and is derived from the physical characteristics of the TF. Practically, this size is often estimated from the observed motifs in the ChIP-seq peaks of the TF. For each j, $P\left(x_j|binding\right)$ is estimated by counting the frequency of the nucleotides in the j'th position of the observed binding sites which are situated in the ChIP-seq peaks. For a motif of length J, the estimation of this probability distribution is stored in a PWM W as followed:

$$W_{i,j} = \frac{1}{N}\sum_{n\in[N]} 1\left(x_j^{(n)} = i\right)$$

where $x^{(n)}$ is the n'th sequence of the found binding sites, $j \in [J]$ the position in the motif and $i \in [4]$ the nucleotide index of A,C,G and T. To enable comparison between the binding likelihood of TFBS of different length, the use of the normalized form of PWM, the position-specific scoring matrix (PSSM), is more convenient:

$$M_{i,j} = \log_2\left(\frac{W_{i,j}}{b_i}\right) \tag{1}$$

where b_i is the prior background model, which is 0.25 in case of nucleotides. From a generative model point of view, the TFBS sequence is generated by an emission model of the PWM. When a correlation is applied on M and the one-hot encoding of the sequence (see figure 3), the result is the log likelihood of a TF binding to a sequence relative to a random sequence.

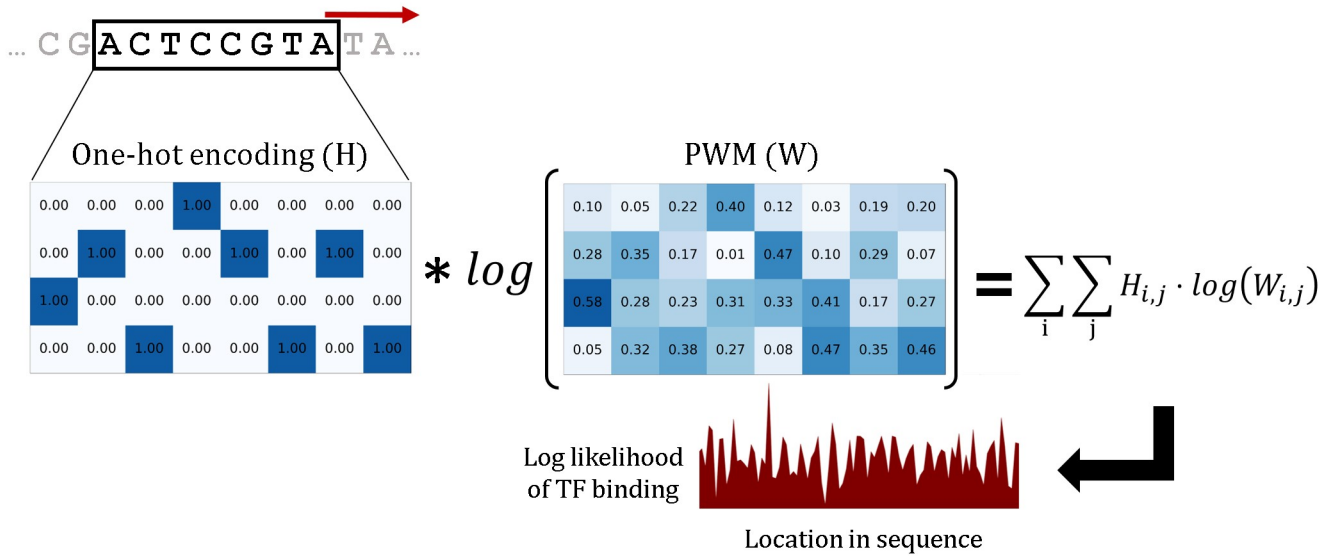


Figure 3: Calculation of a PWM log likelihood. The 2-dimensional correlation between the one-hot encoding of the DNA sequence and the log of the PWM matrix results in the log likelihood of the TF binding at every location in the sequence.

Detection of enhancers and of the tissues in which they are active has been the subject of much research in the last few decades. Specifically, an enhancer detection method relying only on their sequences and without need for biological experimentation is an especially sought-after goal. Such biological experiments, some of which are mentioned in this work, involve cells whose enhancers would need to activate their target genes during the experiment, which is usually an expensive and non-trivial requirement. All methods for detecting active enhancers "in the act" are inherently limited to the specific tissues we can extract and isolate in a lab. Furthermore, many enhancers are only active in specific cell types and at specific stages, and achieving a study of every cell type at every possible stage in complex organisms is not a practical requirement for the foreseeable future. On the other hand, the genome of organisms can be easily and inexpensively sequenced for later analysis *in silico*. The ultimate goal of an efficient computational method which would predict and explain the functional nature of an enhancer sequence has produced positive, yet far from sufficient results over the last years, as reviewed by Kleftogiannis et al. (2016).

As an alternative, a potential way of detecting enhancers only by addressing their sequences, would consist in finding non-coding regions which are conserved across species. Conserved non-coding elements (CNE) have a tendency to reside in clusters, which usually have low gene density but are located in vicinity to genes (Pennacchio et al., 2006). Evidently, the overlap between CNEs and enhancers is imprecise. Some verified enhancers are weakly (or not) conserved between species (Friedli et al., 2010; Rosin et al., 2013; Taher et al., 2011; Lindblad-Toh et al., 2011) and some highly conserved areas in the mouse genome are not associated with regulatory activity, but their deletion still yields viable mice (Ahituv et al., 2007). Nevertheless, an assay of over 200 bp elements with sequence identity of 100% between human and mouse found that less than half showed enhancer activity in mice (Bejerano et al., 2004; Visel et al., 2008). The ultra-conservation of 200 bp enhancer sequences containing TFBSs that are usually shorter than 15 bp raises the possibility that these conserved iter-TFBS parts play a role which it is not yet fully clear.

Almost all cells in every organism contain their entire genomic payload, but only part of this genome is active in any specific cell. Essentially, cells of different type and state differ by gene expression patterns. The reason for this difference between cells lays in regulation components not included in

the Watson and Crick model of the DNA sequence. The location and presence of TFBS, background nucleotides distribution and other sequence-related properties are not sufficient to explain the regulatory role of certain regions in the genome.

Several epigenetic features, correlate with enhancer regions in the genome:

- Accessibility
- TF & cofactors binding
- Histone modifications
- DNA methylation

These mechanisms have measurable features that can be added as a data layer, on top of the genome. Their combination is the main source of identification and prediction for enhancer regions in the genome. Every cell has its own epigenetic features at any element of the genome, often in binary form (e.g. it is either accessible or not, methylated or not). When the epigenetic properties of several cells are measured, a frequency or count of the measured features per DNA locus is usually calculated along the reference genome (see figure 5). The epigenetic data are commonly further processed by calculating their p-value compared to a local environment, to which peak boundaries are determined (peak calling) using algorithms such as MACS2 (Zhang et al., 2008).

In eukaryotes, the DNA is packed around a structure of 8 histone proteins called a nucleosome, forming together a chromatin complex. Similarly to the TFs, the nucleosome binding location in the DNA sequence is not arbitrary. Like them, it has a tendency for specific DNA binding sites (Cutter and Hayes, 2015). The DNA wrapped around a nucleosome has a lesser likelihood of interaction with proteins, because it is physically inaccessible. Accessibility enables the TFs and other proteins to bind to the DNA molecule, hence the enhancer, the promoter and the gene all need to be accessible for a successful transcription to occur. DNase-I hypersensitive (DHS) sites are regions of the DNA which are sensitive to cleavage by the DNase-I enzyme. In these regions the DNA loses the nucleosome, and becomes accessible and therefore potentially active. Measurement of DHS cleavages is available through DNase-seq (Boyle et al., 2008), a high-throughput method for measuring the accessibility epigenetic data of the DNA, usually with a better resolution than histone modifications measurements. A faster and more sensitive technique for accessibility measurement is called ATAC-seq (Visel et al., 2008), and is currently more commonly used.

Histone modifications, also called histone marks or chromatin modifications, are chemical alterations which happen to the long tail-like section of the histone protein. Histones are numbered from 1 to 8, and for example, the acetylation of the lysine amino-acid situated in 14th position in the protein of the 3rd histone is abbreviated as H3K14ac. Along many roles in the cell, such as DNA repair and mitosis, histone modifications have a function in the gene regulation processes. In the past 20 years, a substantial body of research has shown that histone modifications are predictive of enhancer position and activity status (Visel et al., 2007; Heintzman et al., 2009; Galperin and Fernández-Suarez, 2012). The histone modifications are considered to form a certain "histone code" along the genome, which encodes complex information underlying the genomic code and is connected to transcription regulation and other aspects. Compared to other epigenetic information, chromatin modifications have a shorter time scale ranging from seconds to hours (Hayashi-Takanaka et al., 2011), and are therefore considered as being related to the dynamic changes of the cell.

Measurement of histone modifications is also performed using the ChIP-seq method, similarly to the TF binding detection described above. In histone ChIP-seq, antibodies bind to the modifications in the

histone tails (and not to the TF proteins). H3K4me1 and H3K27ac are among the predominant histone modifications of active enhancers; H3K4me1 is enriched on transcribed genes and enhancers prior to activation (Calo and Wysocka, 2013), and is thought to precede the H3K27ac modification (Creyghton et al., 2010; Rada-Iglesias et al., 2011; Zentner et al., 2011) which is known to occur during activation. Other histone modifications present on active enhancers and used for their detection are H3K9ac (Ernst et al., 2011; Karmodiya et al., 2012; Zentner et al., 2011) and H3K18ac (Jin et al., 2011). Even though H3K27ac has been identified as an important marker for the differentiation of active enhancers from poised enhancers (Creyghton et al., 2010), it is not sufficient by itself since when present alongside H3K4me3 it is also an indication for active promoters (Heintzman et al., 2007). In contrast, absence of H3K27ac and enrichment of H3K4me1 and H3K27me3 are typical of poised enhancers (Creyghton et al., 2010).

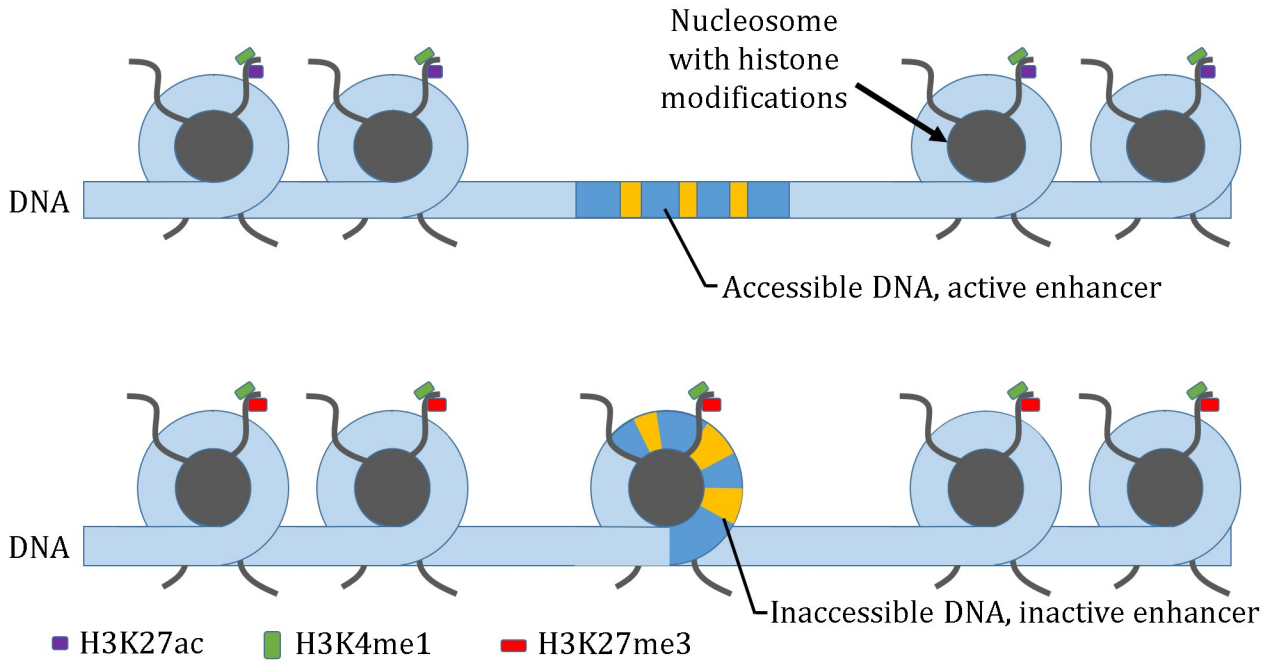


Figure 4: The accessibility of the enhancer and its surrounding histone modifications are connected to its regulatory activity state. The upper diagram shows an active enhancer sequence accessible to the protein interaction needed for transcription, whereas the lower one shows an inactive enhancer wrapped around a nucleosome and therefore inaccessible.

DNA methylation of cytosine nucleotides and cytosine guanine nucleotides pairs (CpG) has been involved in long-term genome silencing in multiple processes (Jones, 2012) and in cell aging (Przybilla et al., 2012). It has been documented as widely correlated with inhibition of gene expression when present in promoters (Tate and Bird, 1993). In enhancer elements, an anti-correlation was found between DNA methylation density and enrichment of active enhancer histone modifications, and TF binding (Stadler et al., 2011; Thurman et al., 2012), although the causes and consequences relationships underlying these correlations are not yet clear. Currently, the most accurate method for wide-scale prediction of the loci of enhancer sequences in a genome is the analysis of histone modifications, and TF and cofactors presence using ChIP-seq from a cell line or from a tissue, combined with DNase-I hypersensitivity (DHS).

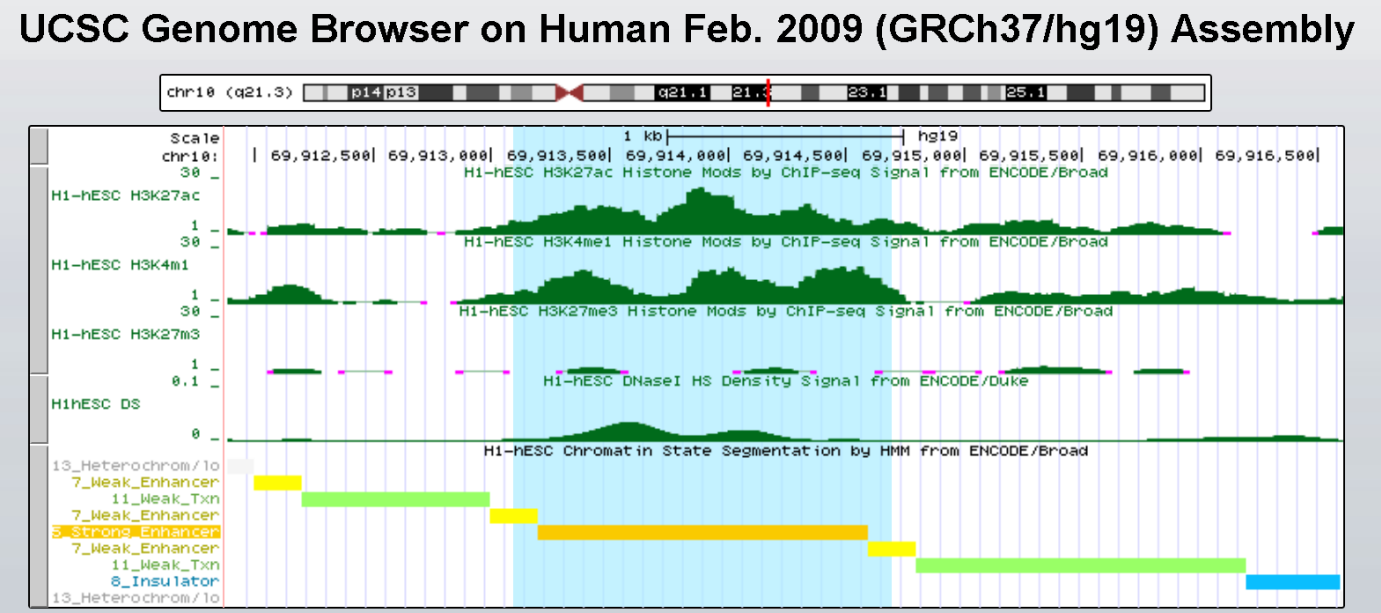


Figure 5: UCSC Genome Browser showing epigenetic features of the 10th chromosome of a H1-hESC cell line. Highlighted in light blue, the peaks of H3K27ac (1st green plot) and H3K4me1 (2nd green plot) histone modifications and the DNase-I hypersensitivity features (4th green plot), together with the absence of H3K27me3 (3rd green plot) signal indicate an active enhancer, as also marked by the ChromHMM classification (bottom). Note that the decrease between the two peaks of H3K27ac and H3K4me1 is located on top of the increase of the DNase-I hypersensitivity, which implies a cleavage between two histone modifications. Taken from [`https://genome-euro.ucsc.edu/cgi-bin/hgTracks`].

1.2 Related Work

The advent of automated DNA sequencing techniques in the 1990s, caused a surge of genomic databases, with the most notable Human Genome Project among them. This great increase of available data led to the development of many computational methods for the analysis of genetic sequences. The main focus of research of these years was on the prediction of genes location, expression and structure using techniques such as hidden Markov model (HMM), decision trees, and dynamic programming as reviewed by Claverie (1997). In the 2000s new techniques for sequence-based high-resolution, genome-scale emerged and resulted in large epigenetic datasets that contributed to the understanding of gene regulation and other interactions between the genome and its environment. In the last few years, several significant computational efforts were made for the prediction of the epigenetic and regulatory properties of DNA elements based on the genetic sequence alone. DeepSEA (Zhou and Troyanskaya, 2015) used a deep convolutional neural network (DCNN) which receives an input of 1000 bp sequence, and outputs a prediction vector of 919 binary features representing the chromatin modifications of 200 bp in the center of the input sequence. The training labels used were the chromatin modifications extracted from ENCODE and Roadmap epigenetic data releases. Basset (Kelley et al., 2016) also used DCNN on the same data, with known PWMs as weight initialization, to predict a binary vector representing accessibility in 164 cell types, based on 600 bp DNA sequences. In DeepBind (Alipanahi et al., 2015) a DCNN was used to predict the binding of 538 TFs and 194 RNA-binding proteins from DNA sequences of varying lengths, based on data from *in vitro* assays. In gkm-SVM (De Beer et al., 2014), gapped *k*-mers presence indicator vectors were used as features for a SVM classifier in order to predict the p300 ChIP-seq signal of DNA sequences of varying length. ChromHMM (Ernst and Kellis, 2012) is a widely used software which tackles the problem of analyzing the epigenetic data to predict the role of fragments of genomic sequence. The algorithm converts the input data of chromatin modification values to a binary form by whether or

not they exceed a defined threshold. The binary input is then inserted to an HMM that classifies the genome states. The disadvantage of these methods is their need for training data of known regulatory elements or epigenetic data which are commonly obtained from GWAS surveys, such as those that were done on 127 obtained human cell types in the Roadmap and ENCODE projects (Kundaje et al., 2015; Ernst et al., 2011).

When a DNA sequence is read from a tissue sample, it is often stored as a sequence of the characters A,C,G and T in FASTA format. For an algorithm to process it, these characters are mapped into a data structure of integers 1,2,3 and 4 respectively. For many algorithms, such as in DeepSEA, Basset, and our HOP-HMM, it is preferable to encode these sequences of integers as a sequence of one-hot vectors (also called indicator vectors), as described in figure 3. A commonly used feature extraction technique of DNA sequences is to represent them as vectors of their in-sequence k-mer frequencies, as used in gkm-SVM. With this technique, the structure in the order of the k-mers is sacrificed for a more meaning-oriented data encoding of fixed length, similarly to the bag-of-words technique in text analysis and natural language processing.

1.3 Classification of Sequences

The goal of machine learning classification models is to arrive from the observed X to its label Y. In the DNA classification case discussed in this work, the goal is to decide the role Y of a DNA sequence X. There are two main approaches to this goal: generative models and discriminative models. Both approaches assume observed variables X and target variables Y, also commonly referred to as data samples and labels.

- Generative models assume a joint probability $P(X,Y)$. Using dataset of (x,y) pairs, one can estimate the distribution $P(X,Y)$, then estimate $P(Y|X)$. The distinctive feature of these models is their ability to generate random instances of the data, either as pairs of (x,y) or as instances of x given y.
- Discriminative models assume a conditional probability $P(Y|X)$, which is estimated directly from the dataset.

Both models eventually base their classifications upon the $P(Y|X)$ estimation. Namely, classifying a data sample x by the most likely label:

$$y_{est} = argmax_y P(Y = y|X = x)$$

Discriminative models are more widely used than generative models, and they are often easier to use and build since they require fewer assumptions on the origin and generation of the data. For instance, the deep neural network (DNN) is a model that has gained much interest lately in the machine learning field, and has also been used for the task of DNA sequences classification. As a discriminative model it assumes very little regarding the way the DNA sequence is generated based on its role, but instead finds features in the sequence which imply its role. Hence it is often difficult to use such a model for later understanding of the nature of the data generation process, or to generate new data from it.

Markov model (Markov, 1906), named after the Russian mathematician Andrey Markov, is a stochastic model which is applied to a system that changes randomly, such as the weather or car traffic. This model is at one of m states $\{S_1,...,S_m\}$ at any time, with the first state being sampled from a distribution $\pi_i = P(y_1 = S_i)$ and the probability distribution of transitions between the states being denoted by

$T_{i,j} = P\left(y_t = S_i | y_{t-1} = S_j\right)$. The travel of the model over the states is named a Markov process, and the sequence of the states visited in the process is called a Markov chain.

The likelihood of a Markov chain X generated by a Markov Model $\theta = \{\pi, T\}$ is a joint probability of the first state and of all following transitions which, due to the independence between transition events, can be written as:

$$\mathscr{L}(\theta; X) = P(x_0, x_1, ..., x_L; \theta) = \pi_{x_0} \cdot T_{x_0,x_1} \cdot T_{x_1,x_2} \cdot ... \cdot T_{x_{L-1},x_L}$$

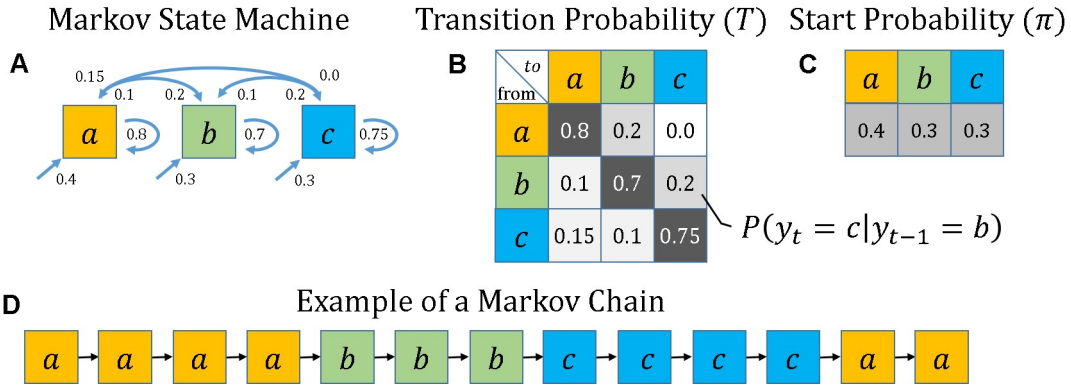


Figure 6: Markov model and **A)** Markov model with 3 states (a, b and c). **B,C)** The model starts with a state sampled from the start distribution π, and travels between the states with a transition probability matrix T. **D)** The model can generate Markov chains of states, where the transition between the states is only conditioned by the previous state, causing the Markov process to be memoryless.

The HMM is a Markov model extension which models a system that travels over hidden states as a Markov process, and while doing so emits variables named observed variables. Like the Markov model, HMM is a generative model, and therefore assumes the existence of a joint probability $P(x_{1:L}, y_{1:L})$ derived from the compact parameters θ. HMM relies on the assumption that the observed DNA sequence $X = x_1, ..., x_L$ is generated by a parameterized model θ, and has a hidden sequence $Y = y_1, ..., y_L$ that was generated alongside it. In this generation process, a single observed variable is emitted for every step of the model, and thus the observed sequence is generated with the same length as the hidden Markov chain. For an alphabet of variables $\{V_1, ..., V_n\}$, and hidden state space $\{S_1, ..., S_m\}$, the observed variable x_t is sampled from an emission distribution conditioned on the hidden state of the model $E_{i,j} = P\left(x_t = V_j | y_t = S_i\right)$. Similarly to the Markov model, the distribution to the first hidden state is marked as π and the transition distribution is marked as T.

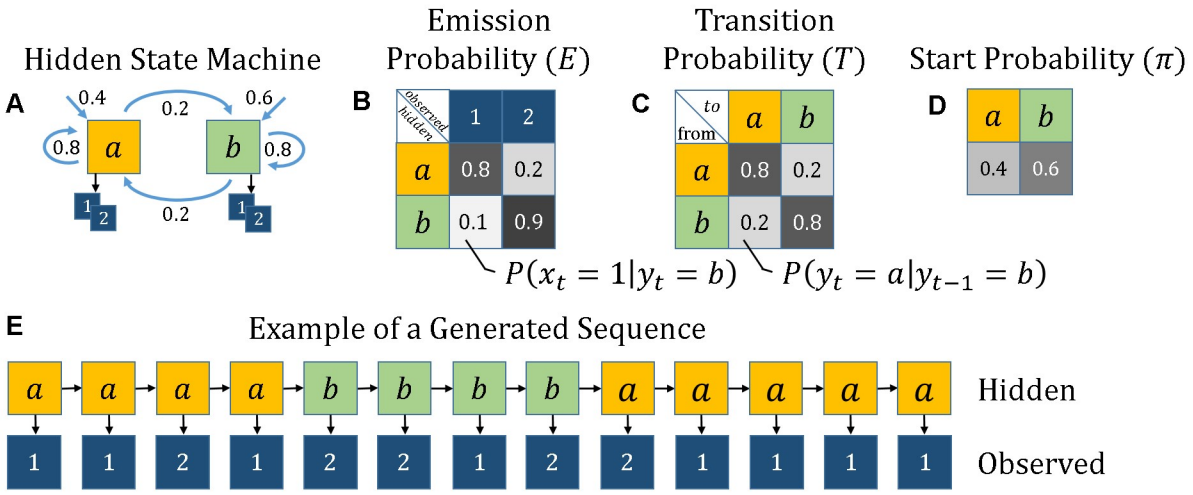


Figure 7: HMM with 2 hidden states. **A,B)** The observed variables (dark blue) are emitted by the hidden state at their location, sampled from the discrete conditional distribution E. **C,D)** The hidden states (yellow and green) behave as Markov model states with start and transition distributions. **E)** The output of the model is an observable sequence with an underlying hidden sequence. The hidden sequence is a Markov chain, where the hidden state emits a single observed variable at each step.

HMM is a very popular signal processing algorithm that has been adopted in the various fields of computational biology since the 1980s. HMM was proposed by Leonard Baum (Baum and Petrie, 1966) and is used for modeling regions with alternating frequencies of patterns and symbols. In a non-biological context, it was used extensively in various engineering fields, especially in speech recognition (Rabiner and Juang, 1993), handwriting recognition (Hu et al., 1996) and digital communication (Turin and Sondhi, 1993).

For example, in the case of an observable DNA sequence, a simplistic model can assume that the sequence is composed of 4 states: genes, promoters, enhancers and background regions. Each of these states would have a different nucleotide frequency, and we assume that the DNA sequence was generated by an HMM with underlying sequences of 4 hidden states, one for each region type. The emitted DNA sequence $x_{1:L}$ is determined by the underlying hidden sequence $y_{1:L}$ which describes the "mode" of the sequence for each location.

Having an HMM with θ on hand and given an observed sequence $x_{1:L}$, two questions arise:

- What is the likelihood that $x_{1:L}$ was generated by the HMM with parameters θ or $P(x_{1:L};\theta)$?
- What is the probability of a hidden state at every location or $P(y_t = j|x_{1:L};\theta)$?

The two above-mentioned probabilities are named the likelihood function and the posterior probabilities of HMM. As in many generative models, HMM likelihood function $\mathscr{L}(\theta|x_{1:L})$ relating to the first question can be split by the total probability law to the sum of all possible hidden sequences:

$$\mathscr{L}(\theta;x_{1:L}) = P(x_{1:L};\theta) = \sum_{y_{1:L}\in[m]^L} P(x_{1:L},y_{1:L};\theta) \tag{2}$$

The probability $P(x_{1:L};\theta)$ is called the incomplete data likelihood function and the probability $P(x_{1:L},y_{1:L};\theta)$ is called the complete-data likelihood function.

In the case of HMM with parameters θ, the complete-data can be calculated by:

$$P(x_{1:L}, y_{1:L}; \theta) = P(y_1; \theta) \cdot P(x_1|y_1; \theta) \cdot \prod_{i=2}^{N} P(y_i|y_{i-1}; \theta) \cdot P(x_i|y_i; \theta) = \pi_{y_1} E_{y_1,x_1} \prod_{i=2}^{L} T_{y_{i-1},y_i} E_{y_i,x_i} \quad (3)$$

Although the computation of the complete-data likelihood of θ in (3) is linear-by-L, naively computing the incomplete data likelihood as in (2) involves the summation of all possible hidden sequences, an impracticable exponential-by-L operation. A dynamic programming approach to overcome this gap uses the Markovian memorylessness of HMM, and answers both the likelihood and the posterior questions raised above. This approach is called the forward-backward algorithm: it was suggested as a step in the Baum-Welch algorithm (Baum and Petrie, 1966), which is an expectation-maximization (EM) algorithm used to find an unknown θ given an observed sequence, and will be described further in a later section. In the forward-backward algorithm, two matrices of size $m \times L$ are calculated, holding the probabilities:

$$\alpha_{j,t} = P(y_t = j, x_{1:t}; \theta)$$
$$\beta_{j,t} = P(x_{t+1:L}|y_t = j; \theta)$$

1.4 Forward Algorithm

The forward probabilities matrix α holds the probability that a sequence $x_{1:t}$ was emitted and that the hidden sequence ended with the state j:

$$\alpha_{j,t} = P(y_t = j, x_{1:t}; \theta)$$

It is calculated by the dynamic programming:

Algorithm 1 Forward Algorithm

Input:

$x_{1:L}$ - Observed DNA sequence

Algorithm:

$for\ j = [1, ..., m]:$

$\alpha_{j,1} = \pi_j \cdot E_{j,x_1}$

$for\ t = [2, ..., L]:$

$for\ j = [1, ..., m]:$

$\alpha_{j,t} = \sum_{j' \in [m]} \alpha_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_t}$

The building of the table is based on the HMM basic assumptions that each hidden state y_t is dependent only on the previous one y_{t-1} and that each observed variable x_t is dependent only on the hidden state that emitted it, y_t.

$$\alpha_{j,t} = P(y_t = j, x_{1:t}; \theta) = P(x_t | y_t = j, x_{1:t-1}; \theta) \cdot P(y_t = j, x_{1:t-1}; \theta) =$$
$$= P(x_t | y_t = j; \theta) \cdot \sum_{j' \in [m]} P\left(y_t = j | y_{t-1} = j'; \theta\right) \cdot P\left(y_{t-1} = j', x_{1:t-1}; \theta\right) =$$
$$= E_{j,x_t} \cdot \sum_{j' \in [m]} T_{j',j} \cdot \alpha_{j',t-1}$$

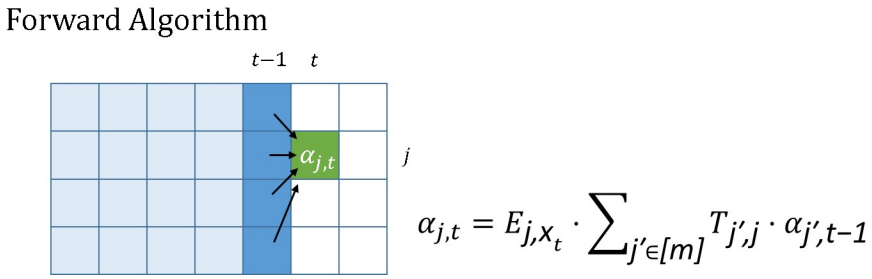


Figure 8: The forward algorithm calculations. The probability stored in $\alpha_{j,t}$ is calculated with the $\alpha_{j',t-1}$ values of the previous steps.

1.5 Backward Algorithm

The backward probabilities matrix β holds the probability that a sequence $x_{t+1:L}$ was emitted given the hidden state at position t had value j:

$$\beta_{j,t} = P(x_{t+1:L} | y_t = j; \theta) \tag{4}$$

It is calculated by the dynamic algorithm:

Algorithm 2 Backward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

$\beta_{1:m,L} = 1$
$for\ t = [L-1, ..., 1]:$
  $for\ j = [1, ..., m]:$
    $\beta_{j,t} = \sum_{j' \in [m]} \beta_{j',t+1} \cdot T_{j,j'} \cdot E_{j',x_t}$

This matrix building process is similarly explained by:

$$\beta_{j,t} = P(x_{t+1:L} | y_t = j; \theta) = \sum_{j' \in [m]} P\left(y_{t+1} = j', x_{t+1:L} | y_t = j; \theta\right) =$$

$$= \sum_{j' \in [m]} P\left(x_{t+2:L} | y_{t+1} = j'; \theta\right) \cdot P\left(x_{t+1} | y_{t+1} = j'; \theta\right) \cdot P\left(y_{t+1} = j' | y_t = j; \theta\right) =$$

$$= \sum_{j' \in [m]} \beta_{j',t+1} \cdot E_{j',x_{t+1}} \cdot T_{j,j'}$$

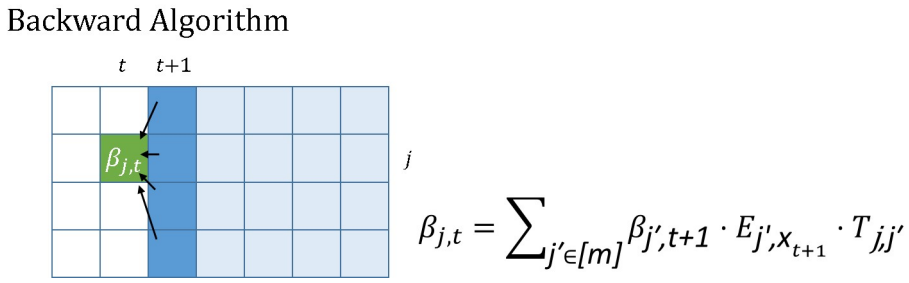


Figure 9: The backward algorithm calculations. The probability stored in $\beta_{j,t}$ is calculated with the $\beta_{j',t+1}$ values of the previous steps.

Once α and β probabilities are obtained, the incomplete data likelihood of HMM can be easily calculated:

$$P(x_{1:L}; \theta) = \sum_{j \in [m]} P(y_L = j, x_{1:L}; \theta) = \sum_{j \in [m]} \alpha_{j,L} \tag{5}$$

And so can the posterior probability:

$$P(y_t = j | x_{1:L}; \theta) = \frac{P(y_t = j, x_{1:L}; \theta)}{P(x_{1:L}; \theta)} = \frac{P(y_t = j, x_{1:t}; \theta) \cdot P(x_{t+1:L} | y_t = j; \theta)}{P(x_{1:L}; \theta)} = \frac{\alpha_{j,t} \cdot \beta_{j,t}}{P(x_{1:L}; \theta)}$$

1.6 Generalized HMM

Although HMM is simple and efficient, applying it to DNA sequences has a major inherited disadvantage: the Markovian lack of memory property. The next state of the model always depends only on the previous state, without further historical consideration. For the task of emitting a TFBS motif where each position has a different emission distribution depending on the location in the motif, an HMM would need to differentiate multiple hidden states according to their positions in the motif. This means that for an HMM to be able to emit even a small number of short motifs, it needs to hold a large number of states which require learning a large number of parameters, e.g. for the ability to emit 50 motifs of length 5, an HMM would need to have over 60,000 parameters (50×5 states and $(50 \times 5)^2$ transitions). Furthermore, the enhancer modeling task at hand is even more complex, since we wish to model multiple enhancers and backgrounds states, each having a different probability of emitting motifs and an unique k-order emission distribution when not in these motifs. For the prior assumption of our data structure, the required number of model parameters would have been about 10^7, large enough to generate problems such as unfeasible memory complexity and overfitting.

A common way to avoid overfitting the data when training machine learning models is to reduce the complexity of the model by fixing some of its parameters. Our proposed HOP-HMM addresses both the memory issue and the overfitting issue, while remaining equivalent to a regular HMM with a large number of states having fixed parameters. Namely, most of the transition probabilities are fixed to zero and therefore never stored in the memory, and some of the emission probabilities are predetermined and

remain fixed during the training. This allows us to train a model with the enhancer prior assumptions of motifs and high-order emission, without overfitting and with reasonable memory complexity.

In a generalized HMM (GHMM), the transition or the emission are sampled from a different distribution type assigned to each of the states in the model. Some of the states in the system may emit zero or multiple observable variables, sampled from custom emission models specifically tailored for the expected scenario. Such models were used for gene prediction in the 1990s (Haussler and Eeckman, 1996; Burge and Karlin, 1997), in which specific exon states emitted codons instead of single nucleotides, and feed forward neural networks were used to evaluate the probability of certain transitions.

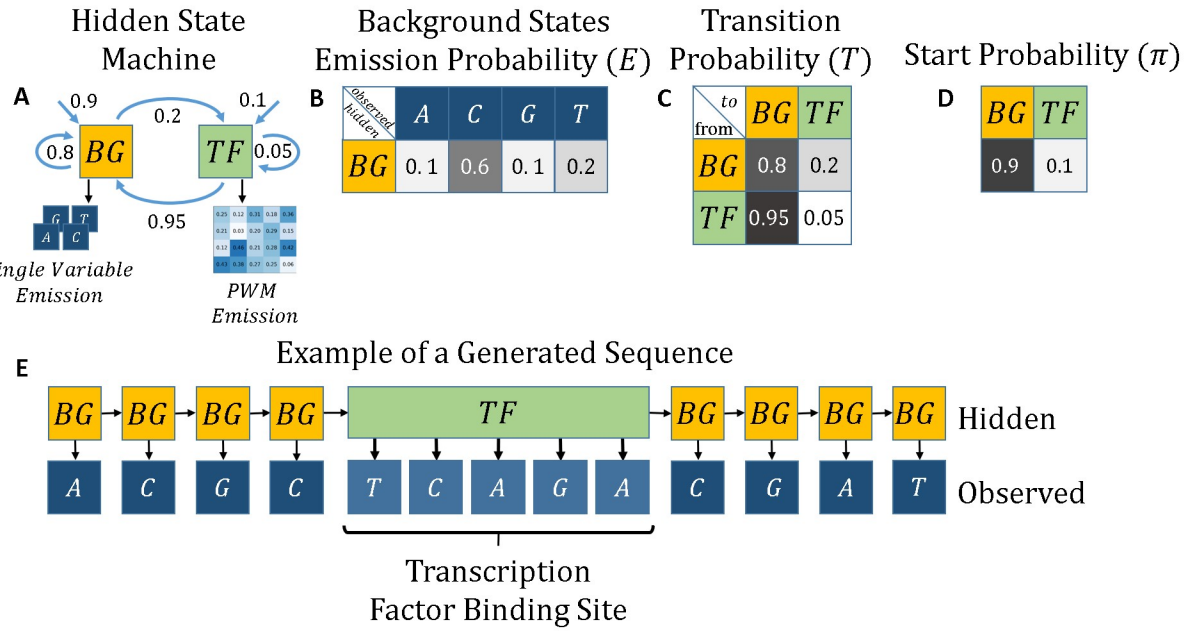


Figure 10: Generalized HMM that can emit TFBSs as described by Kaplan and Biggin (2012) **A)** The GHMM contains a TF state which emits a motif using a PWM. The model has one background hidden state (yellow) and one TF hidden state. **B,C,D)** Although the TF state emits motifs with 5 bp, the rest of the emissions, transitions and start probabilities remain the same as in a regular HMM. **E)** An example output generated from the model, showing the TFBS motif sampled in an arbitrary location inside a sequence.

Another generalization made to the HMM and called high-order HMM uses conditional distribution by making the transition and emission dependent on previous hidden states (Ferguson, 1980; Mari et al., 1997; Du Preez, 1998; Lee and Lee, 2006). Although these HMM variants are capable of expressing a more complex structure of DNA sequence (different k-mers frequencies in the genomic regions), the number of parameters required for DNA analysis tends to rise with the increase of the assumed complexity of the DNA structure. The increase of hidden states needed may introduce overfitting in the learning process, when the data size is limited.

Instead of high-order emission which depends on the previous hidden states, the less researched field of high-order emission depending on previously emitted observable variables was used. Such an HMM variant is better suited to the local span nature of the emission of k-mer structures, but it only requires $O\left(m^2+4^k\right)$ parameters compared to $O\left(m^k\right)$ parameters that would have been required for holding a k-mer distribution in a regular HMM, where m is the number of hidden states of the HOP-HMM, and k is the number of previous states in the dependency.

1.7 HOP-HMM

HOP-HMM is a GHMM well adapted to the structure of regulatory sequences containing TFBSs, due to the TFBS emitting TF states that take part in the generation process of the sequence. In view of the assumed local physical nature of the TF binding of DNA sequences and of the success of HMM in gene prediction, we think the memorylessness of HMMs fits well the task of enhancer prediction. HOP-HMM balances between the Markovian memorylessness and the observed k-mer frequencies of the regulatory regions in the genome.

HOP-HMM extends the GHMM that was introduced by Kaplan and Biggin (2012), where some of the hidden states emit TFBS sampled from PWMs in order to predict enhancer locations in the genome. In HOP-HMM we added a high-order conditioning to the emission probability distribution of the background states.

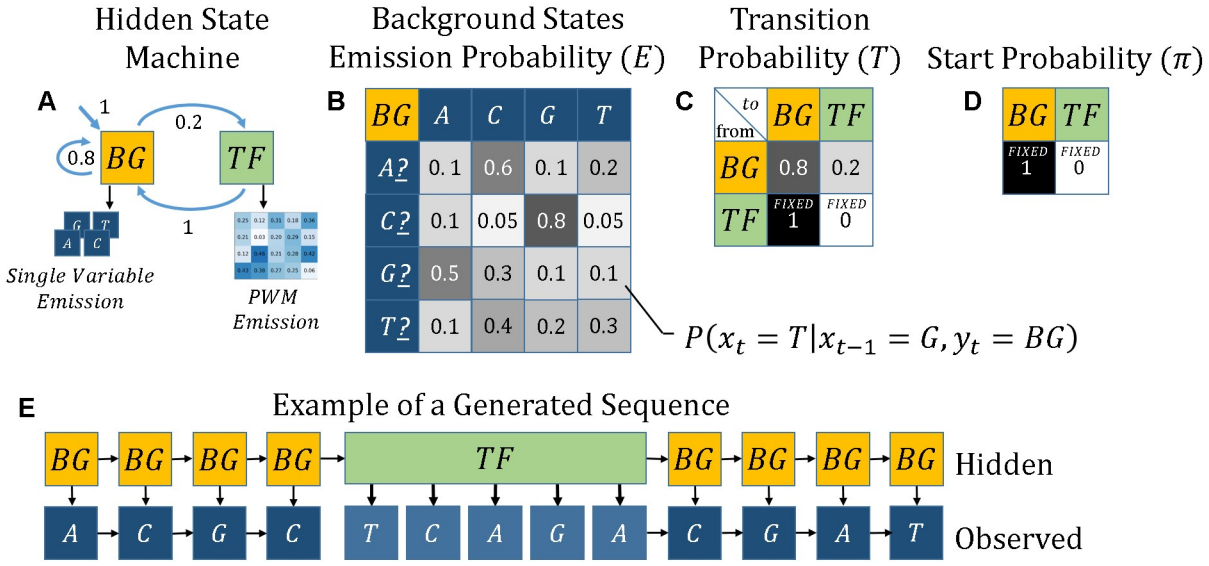


Figure 11: Small HOP-HMM with two states. **A)** The model has one background state that emits single nucleotides, and one TF state that emits a TFBS and returns to the background state. **B)** The background state has second-order emission, meaning its emission probability distribution is conditioned on the previous nucleotide. **C,D)** Unlike GHMM, in HOP-HMM a TF state cannot transition into itself, and cannot be the starting hidden state. **E)** The TF state emits multiple observable variables that represent a TFBS sampled from a PWM.

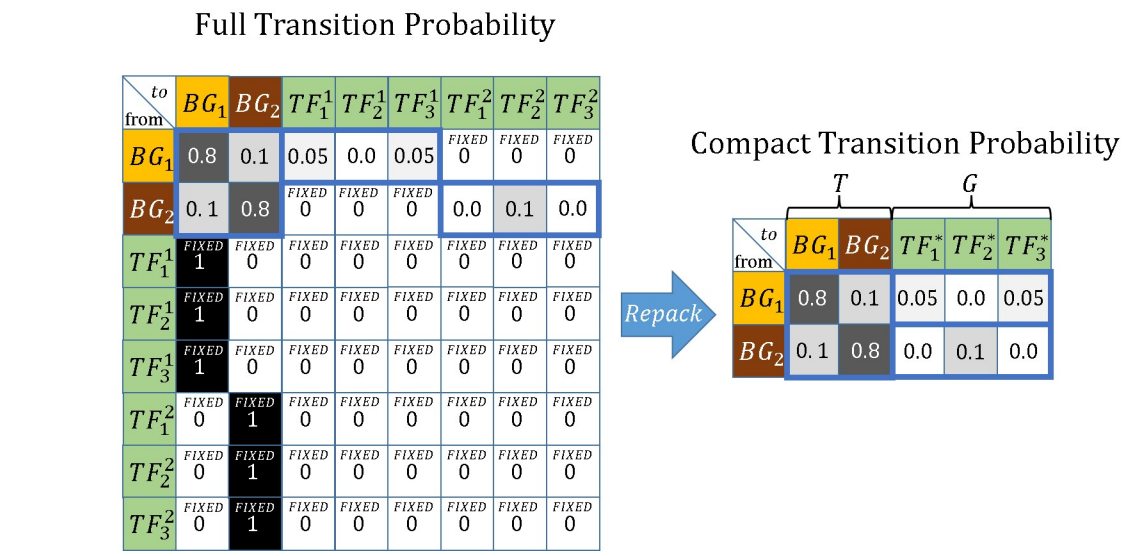


Figure 12: Compact form of HOP-HMM for holding the transition probability distributions. Instead of holding a single sparse 8×8 transition matrix, we hold only the non-fixed transition probabilities, split into T and G matrices. The non-fixed transition probabilities held in the compact form are those between background states, and between background states and their TF states (outlined with blue). The concatenation of a row in T and G holds the probability distribution of the next hidden state given the current background state.

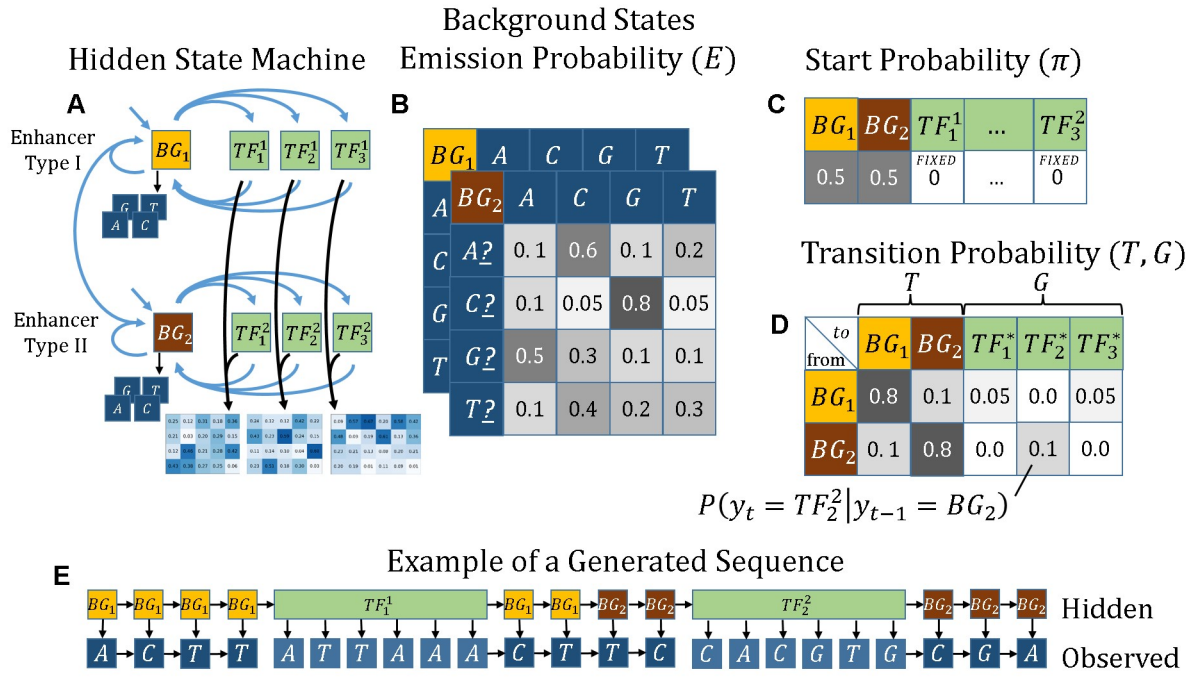


Figure 13: **A)** A more complex HOP-HMM with two background states BG_1 and BG_2, where each has three TF states with three PWMs. **B)** Each of the background states has its own second-order emission distribution in a 4×4 matrix. **C)** The start hidden state distribution π allows only background states to start the hidden sequence. **D)** The transition probability distribution is held by matrices T and G. **E)** The example-generated sequence is built out of two types of sequences, each with its own TFBS frequency and background nucleotide bigram frequency, representing two alternating types of enhancers.

We use two indices to describe a hidden state in HOP-HMM:

- background states are indexed as $(j,0)$ where $j \in [m]$, and m is the number of background states.
- TF states are indexed as (j,l) where $j \in [m]$, $l \in [k]$, and k is the number of TF states which are associated to each of the background states.

For example, in figure 13 we see a HOP-HMM with $m = 2$ and $k = 3$ and a total of 8 hidden states $(2+3\times 2)$. The TF state (j,l) belongs to the $(j,0)$ background state (see figure 14), and the only allowed transfer into (j,l) is from its background state $(j,0)$. Note that we used a simpler BG_j notation in figures 11 and 13 for readability.

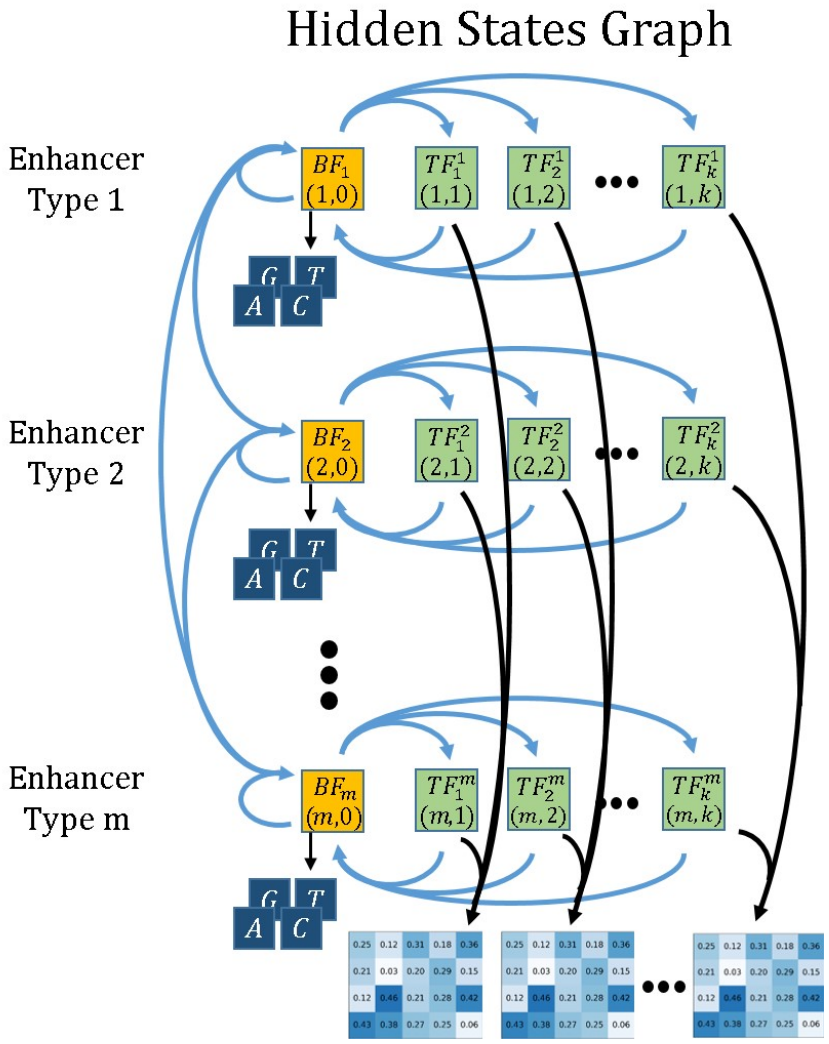


Figure 14: General hidden states graph of HOP-HMM. Each row represents an enhancer type, where each of the m background states (yellow) has k TF states (green). Not all transitions are possible: moving between the rows is possible only through a background-to-background state transition.

While most background states $(j,0)$ represent an enhancer type, we also wish to model true background regions between enhancers, which carry no regulatory role and have no TFBSs. To that end, we predefine one or more background states as non-enhancers by restricting their transfer probability into their TF states, as seen in the results section.

HOP-HMM is defined with k PWMs $W_1, W_2, ..., W_k$ that remain fixed during training. Each of the k PWMs is shared with m TF states, e.g. the PWM W_l, where $l \in [k]$ is shared between subs-states $(1,l),(2,l),...,(m,l)$ and is used for the TF state emission sampling. The PWMs vary in their column amounts (as the different TFBSs vary in length), and each column represents a nucleotide distribution at that position. When the model enters a TF state, it emits a motif by sampling independently from a PWM column by column, as described in figure 10.

The background states, denoted as $(1,0),(2,0),...,(m,0)$, are responsible for the emission of inter-TFBS parts of the enhancers lacking long motifs. Similarly to regular states in HMM, background states emit single nucleotides whose emission is conditional on the previous nucleotides emitted in the DNA sequence. The emission from background states is performed by sampling a nucleotide from the distributions stored in E array. E dimension is $o+1$, its size is $m \times 4 \times 4 \times ... \times 4$ (with o fours) and its values describe the emission probability distribution $E_{j,x_{t-o+1},x_{t-o+2},...,x_t} = P(x_t | y_t = (j,0), x_{t-o+1}, ..., x_{t-1})$, meaning that when x_t is sampled by the model, the preceding $o-1$ observed variables are used as indices of the array to obtain the emission probability distribution vector $E_{j,x_{t-o+1},x_{t-o+2},...,x_{t-1},*}$.

For the first variables emitted in the sequence, the missing dimensions of the preceding variables are summed to form the probability distribution vector, e.g. at position $t = o-1$, a single variable is missing to emit x_t and the probability vector that is used instead for the emission is $\sum_{i \in [4]} \frac{E_{j,i,x_1,...,x_{t-1}}}{4}$.

In HOP-HMM, the first hidden state in a sequence can only be a background state. As in HMM, the first background state is chosen by sampling from π, a probability vector $\pi_j = P(y_1 = (j,0))$. Once in a background state, the model can only transit into a small subset of states, and since most of the possible transitions are not allowed, a single transition matrix from all states to all states would be sparse. Instead, as described in figure 12, we only hold the possible transition probabilities in two matrices, representing the two types of allowed transitions:

- T is a $m \times m$ matrix for background-to-background state transitions

$$T_{j_1,j_2} = P(y_{t+1} = (j_2,0) | y_t = (j_1,0))$$

- G is a $m \times k$ matrix for background-to-TF state transitions

$$G_{j,l} = P\left(y_{t+1:t+|W_l|} = (j,l) | y_t = (j,0)\right)$$

When in a background state, after its observable variable emission the model samples its next hidden state from a probability vector which is composed of the concatenation of the row in T and the row in G, both rows are of the index of the background state. If the next hidden state is chosen to be a TF state, it emits its TFBS from the PWM associated with the TF state. After that, the model will return back to the preceding background state, which will emit another observable variable and so on.

2 Methods

2.1 Baum-Welch Algorithm

When fitting an HMM to a DNA sequence, we seek the parameters $\hat{\theta}$ that best explain the sequence via an algorithm called Baum-Welch algorithm, which is a specific case of EM algorithm. Formally, given the observed DNA sequence $x_{1:L}$, we would like to find the parameters that maximize the incomplete data likelihood:

$$\hat{\theta} = argmax_{\theta} \mathscr{L}(\theta | x_{1:L})$$

Even though the incomplete data likelihood of HMM in (2) is derivable by θ, optimizing it is as difficult as calculating it, and is therefore impractical. Instead, the strategy of the EM algorithm is to iteratively optimize the expected value of the complete data log-likelihood $log\left(P\left(x_{1:L}, y_{1:L} | \theta^{'}\right)\right)$ over all possible $y_{1:L}$ where $\theta^{'}$ is the model parameters from the previous EM iteration (or guessed parameters in the first iteration) and while assuming a fixed observed $x_{1:L}$, as it is in the given DNA sequence. For this task, we define our target function Q:

$$Q\left(\theta, \theta^{'}\right) = E_Y\left[log\left(P(X,Y;\theta)\right) | X = x_{1:L}, \theta^{'}\right] = \sum_{y_{1:L} \in [m]^L} log\left(P(x_{1:L}, y_{1:L};\theta)\right) \cdot P\left(x_{1:L}, y_{1:L}; \theta^{'}\right) \quad (6)$$

Here E is expressing an expected value, not to be confused with the HMM emission probability distribution. Every EM iteration is built out of two parts called the E (expectation) step and the M (maximization) step. In the E-step we calculate the probabilities needed for the maximization of Q and in the M-step we infer the θ that maximizes it. We will update θ to the maximum of $Q\left(\theta, \theta^{'}\right)$ in every M-step of the EM algorithm until convergence.

Using (3) we will split the Q function (6) into three independent parts:

$$Q\left(\theta, \theta^{'}\right) = \sum_{y_{1:L} \in [m]^L} log\pi_{y_1} \cdot P\left(x_{1:L}, y_{1:L}; \theta^{'}\right) +$$

$$+ \sum_{y_{1:L} \in [m]^L} \left(\sum_{t \in 2...L} logT_{y_{t-1}, y_t}\right) \cdot P\left(x_{1:L}, y_{1:L}; \theta^{'}\right) + \sum_{y_{1:L} \in [m]^L} \left(\sum_{t \in [L]} logE_{y_t, x_t}\right) \cdot P\left(x_{1:L}, y_{1:L}; \theta^{'}\right)$$

Then by manipulating the summation, the exponential hidden sequence summation could be simplified with the law of total probability to:

$$Q\left(\theta, \theta^{'}\right) = \sum_{j \in [m]} log\pi_j \cdot P\left(x_{1:L}, y_1 = j; \theta^{'}\right) +$$

$$+ \sum_{t \in 2...L} \sum_{j_1, j_2 \in [m]} logT_{j_1, j_2} \cdot P\left(x_{1:L}, y_{t-1} = j_1, y_t = j_2; \theta^{'}\right) + \sum_{t \in [L]} \sum_{j \in [m]} logE_{j, x_t} \cdot P\left(x_{1:L}, y_t = j; \theta^{'}\right)$$

Each of the three parts above is a set of constraint linear functions that could be derived and maximized independently using Lagrange multipliers, and under the following probability distribution constraints:

- $\sum_{j\in[m]} \pi_j = 1$
- $\sum_{j_2\in[m]} T_{j_1,j_2} = 1$ for all $j_1 \in [m]$
- $\sum_{b\in[n]} E_{j,b} = 1$ for all $j \in [n]$

where m is the number of different hidden states and n is the number of different observed variables (4 in our case of DNA).

First, we start with maximizing the first π part using Lagrange multiplier λ:

$$\frac{\partial}{\partial \pi_j}\left(\sum_{j'\in[m]} log\pi_{j'} P\left(x_{1:L}, y_1 = j'; \theta'\right) + \lambda\left(1 - \sum_{j'\in[m]} \pi_{j'}\right)\right) = 0$$

we derive the term and get $\frac{P\left(x_{1:L}, y_1 = j; \theta'\right)}{\pi_j} = \lambda$ for $j \in [m]$. Then we use these m equations to get $\lambda = P\left(x_{1:L}; \theta'\right)$, which yields the reestimated π_j:

$$\pi_j = \frac{P\left(x_{1:L}, y_1 = j; \theta'\right)}{P\left(x_{1:L}; \theta'\right)} = P\left(y_1 = j | x_{1:L}; \theta'\right) \tag{7}$$

Then, we define a Lagrange multiplier λ_{j_1} for each $j_1 \in [m]$ for the T part:

$$\frac{\partial}{\partial T_{j_1,j_2}}\left(\sum_{t\in 2\ldots L} logT_{j_1,j_2} \cdot P\left(x_{1:L}, y_{t-1} = j_1, y_t = j_2; \theta'\right) + \lambda_{j_1}\left(1 - \sum_{j'\in[m]} T_{j_1,j'}\right)\right) = 0$$

which yields $\lambda_{j_1} = \frac{\sum_{t\in 2\ldots L} P\left(x_{1:L}, y_{t-1} = j_1, y_t = j_2; \theta'\right)}{T_{j_1,j_2}}$ for $j_2 \in [m]$

and when all m equations are summed, gives $\lambda_{j_1} = \sum_{t\in 2\ldots L} P\left(x_{1:L}, y_{t-1} = j_1; \theta'\right)$

Therefore the update of T_{j_1,j_2} will be:

$$T_{j_1,j_2} = \frac{\sum_{t\in 2\ldots L} P\left(x_{1:L}, y_{t-1} = j_1, y_t = j_2; \theta'\right)}{\sum_{t\in 2\ldots L} P\left(x_{1:L}, y_{t-1} = j_1; \theta'\right)} = \frac{\sum_{t\in 2\ldots L} P\left(y_{t-1} = j_1, y_t = j_2 | x_{1:L}; \theta'\right)}{\sum_{t\in 2\ldots L} P\left(y_{t-1} = j_1 | x_{1:L}; \theta'\right)} \tag{8}$$

Finally, we will define a Lagrange multiplier λ_j for every $j \in [m]$ for the E part:

$$\frac{\partial}{\partial E_{j,b}}\left(\sum_{t\in[L]} logE_{j,x_t} \cdot P\left(x_{1:L}, y_t = j; \theta'\right) + \lambda_j\left(1 - \sum_{b'\in[n]} E_{j,b'}\right)\right) = 0$$

This step is slightly trickier due to the derivation of $\frac{\partial E_{j,x_t}}{\partial E_{j,b}} = 1_b(x_t)$ where $1_b(x_t) = \begin{cases} 1 & b = x_t \\ 0 & otherwise \end{cases}$.

We get $\lambda_j = \frac{\sum_{t\in[L]} P\left(x_{1:L}, y_t = j; \theta'\right) \cdot 1_b(x_t)}{E_{j,b}}$ for $b \in [n]$

and when all n equations are summed, gives $\lambda_j = \sum_{t\in[L]} P\left(x_{1:L}, y_t = j; \theta'\right) \cdot 1_b(x_t)$

Therefore the update of $E_{j,b}$ will be:

$$E_{j,b} = \frac{\sum_{t\in[L]} P\left(x_{1:L}, y_t = j; \theta'\right) \cdot 1_b(x_t)}{\sum_{t\in[L]} P\left(x_{1:L}, y_t = j; \theta'\right)} = \frac{\sum_{t\in[L]} P\left(y_t = j | x_{1:L}; \theta'\right) 1_b(x_t)}{\sum_{t\in[L]} P\left(y_t = j | x_{1:L}; \theta'\right)} \tag{9}$$

In order to be able to calculate these reestimations of θ as written in (7), (8) and (9), we still need to calculate the two probability terms they contain. To resemble the notations coined by Rabiner (1989), the first widely accepted HMM application, we will denote these as γ and ξ

$$\gamma_{t,j} = P\left(y_t = j | x_{1:L}; \theta'\right) \tag{10}$$

$$\xi_{t,j_1,j_2} = P\left(y_{t-1} = j_1, y_t = j_2 | x_{1:L}; \theta'\right) \tag{11}$$

We will use (5) and the output of the forward-backward algorithm α and β for their calculation:

$$\gamma_{t,j} = \frac{P\left(y_t = j, x_{1:L}; \theta'\right)}{P\left(x_{1:L}; \theta'\right)} = \frac{P\left(y_t = j, x_{1:t}; \theta'\right) \cdot P\left(x_{t+1:L} | y_t = j, x_{1:t}; \theta'\right)}{P\left(x_{1:L}; \theta'\right)} =$$

$$= \frac{P\left(y_t = j, x_{1:t}; \theta'\right) \cdot P\left(x_{t+1:L} | y_t = j; \theta'\right)}{P\left(x_{1:L}; \theta'\right)} = \frac{\alpha_{j,t} \cdot \beta_{j,t}}{\sum_{j\in[m]} \alpha_{j,L}}$$

$$\xi_{t,j_1,j_2} = \frac{P\left(y_{t-1} = j_1, y_{t]} = j_2, x_{1:L}\right)}{P\left(x_{1:L}; \theta'\right)} =$$

$$= \frac{P\left(y_{t-1} = j_1, x_{1:t-1}; \theta'\right) \cdot P\left(y_t = j_2 | y_{t-1} = j_1; \theta'\right) \cdot P\left(x_t | y_t = j_2; \theta'\right) \cdot P\left(x_{t+1:L} | y_t = j_2; \theta'\right)}{P\left(x_{1:L}; \theta'\right)} =$$

$$= \frac{\alpha_{j_1,t-1} \cdot T_{j_1,j_2,} \cdot E_{j_2,x_t} \cdot \beta_{j',t}}{\sum_{j\in[m]} \alpha_{j,L}}$$

The calculation of α, β, γ and ξ matrices is considered the E-step of the Baum-Welch algorithm and allows us to update θ in the M-step and finish the EM iteration.

2.2 Baum-Welch Algorithm Adaptation

The transition and emission mechanisms of HOP-HMM are different and therefore the complete data likelihood of HOP-HMM requires a different calculation for the Baum-Welch algorithm to hold. The Baum-Welch algorithm can be adjusted to infer the parameters of the HOP-HMM variant $\theta = \{\pi, E, G, T\}$ from a DNA sequence X. As in the regular Baum-Welch algorithm covered in the previous section, given a sequence X at each EM iteration we optimize the Q function in (6):

$$
\begin{aligned}
Q\left(\theta,\theta^{'}\right) = & \sum_{j\in[m]} log\pi_j \cdot P\left(x_{1:L}, y_1 = (j,0); \theta^{'}\right) \\
& + \sum_{t\in 2\ldots L} \sum_{j_1,j_2\in[m]} logT_{j_1,j_2} \cdot P\left(x_{1:L}, y_{t-1} = (j_1,0), y_t = (j_2,0); \theta^{'}\right) \\
& + \sum_{t\in 2\ldots L} \sum_{j\in[m],l\in[k]} logG_{j,l} \cdot P\left(x_{1:L}, y_{t-1} = (j,0), y_t = (j,l); \theta^{'}\right) \\
& + \sum_{t\in o,\ldots,L} \sum_{j\in[m]} logE_{j,b_1,\ldots,x_t} \cdot P\left(x_{1:L}, y_t = (j,0); \theta^{'}\right) \\
& + \sum_{t\in[L]} \sum_{l\in[k]} log\mathscr{L}\left(W_l; x_{t:t+|W_l|-1}\right) \cdot P\left(x_{1:L}, y_{t:t+|W_l|-1S} = (j,l); \theta^{'}\right)
\end{aligned}
$$

where $\mathscr{L}(W;\bar{x})$ is the likelihood of the TFBS $\bar{x}$ to be emitted by PWM W: $\mathscr{L}(W;\bar{x}) = P(\bar{x};W) = \prod_{i\in\{1,\ldots,|\bar{x}|\}} W_{\bar{x}_i,i}$.

Note that the last addition component, which holds the TFBS log likelihood, does not contain elements from θ since the PWMs are not learned in HOP-HMM and thus is not reestimated in the M-steps.

Similarly to the process we covered in the regular EM, the θ which optimizes $Q(\theta,\theta')$ is achieved by optimizing its 3 independent parts, each with its own constraint under which we optimize Q are:

- $\sum_{j\in[m]} \pi_j = 1$
- $\sum_{j_2\in[m]} T_{j_1,j_2} + \sum_{l\in[k]} G_{j_1 l} = 1$ for all $j_1 \in [m]$
- $\sum_{b_o\in[n]} E_{j,b_1,\ldots,b_o} = 1$ for all $j \in [n]$

2.2.1 M-step

The π and E conditions produce a very similar maximization as in the regular Baum-Welch (7) and (9):

$$
\pi_j = \frac{P\left(x_{1:L}, y_1 = (j,0)|\theta';\theta'\right)}{P\left(x_{1:L}|\theta';\theta'\right)} = P\left(y_1 = (j,0)|x_{1:L};\theta^{'}\right) \tag{12}
$$

$$
E_{j,b_1,\ldots,b_o} = \frac{\sum_{t\in o,\ldots,L} P\left(x_{1:L}, y_t = (j,0); \theta^{'}\right) 1_{b_1,\ldots,b_o}(x_{t-o+1,\ldots,t})}{\sum_{t\in o,\ldots,L} P\left(x_{1:L}, y_t = (j,0); \theta^{'}\right)} \tag{13}
$$

As for the second condition of T and G, we will define the Lagrange multipliers λ_{j_1} for $j_1 \in [m]$ and derive the two terms that contain T and G:

$$
\frac{\partial}{\partial T_{j_1,j_2}} \left(\sum_{t\in 2\ldots L} logT_{j_1,j_2} \cdot P\left(x_{1:L}, y_{t-1} = (j_1,0), y_t = (j_2,0); \theta^{'}\right) + \lambda_{j_1}\left(1 - \sum_{j'\in[m]} T_{j_1,j'} - \sum_{l\in[k]} G_{j_1 l}\right)\right) = 0
$$

$$\frac{\partial}{\partial G_{j_1,l}}\left(\sum_{t\in 2\ldots L} logG_{j_1,l}\cdot P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_1,l);\theta^{'}\right)+\lambda_{j_1}\left(1-\sum_{j'\in[m]}T_{j_1,j'}-\sum_{l\in[k]}G_{j_1 l}\right)\right)=0$$

which yields $\lambda_{j_1}=\frac{\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_2,0);\theta^{'}\right)}{T_{j_1,j_2}}$ and $\lambda_{j_1}=\frac{\sum_{t\in 2\ldots L}\cdot P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_1,l);\theta^{'}\right)}{G_{j_1 l}}$

for $j_2\in[m]$ and $l\in[k]$. When the $m+k$ equations are summed we receive:

$$\lambda_{j_1}=\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_2,0);\theta^{'}\right)+\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_1,l);\theta^{'}\right)=$$

$$=\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0);\theta^{'}\right)$$

which gives us the updates

$$T_{j_1,j_2}=\frac{\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0),y_t=(j_2,0);\theta^{'}\right)}{\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j_1,0);\theta^{'}\right)}=\frac{\sum_{t\in 2\ldots L}P\left(y_{t-1}=(j_1,0),y_t=(j_2,0)|x_{1:L};\theta^{'}\right)}{\sum_{t\in 2\ldots L}P\left(y_{t-1}=(j_1,0)|x_{1:L};\theta^{'}\right)} \tag{14}$$

$$G_{j,l}=\frac{\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j,0),y_t=(j,l);\theta^{'}\right)}{\sum_{t\in 2\ldots L}P\left(x_{1:L},y_{t-1}=(j,0);\theta^{'}\right)}=\frac{\sum_{t\in 2\ldots L}P\left(y_{t-1}=(j,0),y_t=(j,l)|x_{1:L};\theta^{'}\right)}{\sum_{t\in 2\ldots L}P\left(y_{t-1}=(j,0)|x_{1:L};\theta^{'}\right)} \tag{15}$$

2.2.2 E-step

Preceding the M-step where we update components of θ by (12), (13), (14) and (15), in the E-step we will calculate the three probability terms they contain, denoted as γ, ξ and η:

$$\gamma_{j,t}=P\left(y_t=(j,0)|x_{1:L};\theta^{'}\right) \tag{16}$$

$$\xi_{j_1,j_2,t}=P\left(y_{t-1}=(j_1,0),y_t=(j_2,0)|x_{1:L};\theta^{'}\right) \tag{17}$$

$$\eta_{j,l,t}=P\left(y_{t-1}=(j,0),y_t=(j,l)|x_{1:L};\theta^{'}\right) \tag{18}$$

For the calculation of these probabilities, we first need to calculate the forward and backward probabilities output from a HOP-HMM adjusted forward-backward algorithm. In the forward-backward algorithm adaptation for HOP-HMM, we will only build the probabilities of entering the background states since the TF states probabilities will not be needed in the later parts of the E-step. The adjustments for the forward and backward algorithms lead to the new requirement to support transitions into both background states and TF states.

The forward probabilities for the HOP-HMM are:

$$\alpha_{j,t}=P(y_t=(j,0),x_{1:t})$$

Algorithm 3 Forward Algorithm for HOP-HMM

Input:

X - Observed DNA sequence

Algorithm:

$$
\begin{aligned}
&for\ j = [1,...,m]:\\
&\quad \alpha_{j,1} = \pi_j \cdot E_{j,x_1}\\
&for\ t = [2,...,L]:\\
&\quad for\ j = [1,...,m]:\\
&\quad\quad \alpha_{j,t} = \underbrace{\sum_{j' \in [m]} \alpha_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_{t-o+1},...,x_t}}_{\text{background state transitions}}\\
&\quad\quad\quad + \underbrace{\sum_{l \in [k]} \alpha_{j,t-|W_l|-1} \cdot G_{j,l} \cdot \mathscr{L}\left(W_l; x_{t-|W_l|},...,x_{t-1}\right) \cdot E_{j,x_{t-o+1},...,x_t}}_{\text{TF state transitions}}
\end{aligned}
$$

In the beginning of the sequence, when $1 \leq t < o$, part of the preceding observable variables are missing. Since E has $o+1$ dimensions, $E_{j,x_1,...,x_t}$ is not defined for such locations, so we define it here as:

$$E_{j,x_1,...,x_t} = \sum_{b_1,...,b_{o-t} \in \{A,C,G,T\}} \frac{1}{4^{o-t}} \cdot E_{j,b_1,..,.b_{o-t},x_1,...,x_t}$$

We use the fact that $P(A) = \sum_{b \in B} P(b) \cdot P(A|b)$ and the assumption that the observable variables preceding the sequence came from a uniform distribution. In addition, when summing the TF state transition part, PWMs with a length equal or bigger than $t+1$ are out-of-sequence TFBSs and therefore are not part of the summation.

The backward probabilities for the HOP-HMM are:

$$\beta_{j,t} = P\left(x_{t+1:L} | y_t = (j,0), x_{1:t}\right)$$

The noticeable difference between the backwards probabilities of HMM in 4 and those are the dependency on $x_{1:t}$. In a regular HMM, this dependency is insignificant because the emissions after t are dependent on the hidden state y_t and not on the previous variables, meaning $P\left(x_{t+1:L} | y_t = j, x_{1:t}\right) = P\left(x_{t+1:L} | y_t = j\right)$. On the other hand, this is not the case in HOP-HMM since the emission of the background states has high-order conditioning and is therefore dependent on the previous variables. After describing the adjustment of the backward algorithm, we will show that this adjustment leads to the required terms of the M-step of the Baum-Welch algorithm.

Algorithm 4 Backward Algorithm for HOP-HMM

Input:
X - Observed DNA sequence
Algorithm:
$\beta_{1:m,L} = 1$
$for\ t = [L-1, ..., 1]:$
$for\ j = [1, ..., m]:$

$$\beta_{j,t} = \underbrace{\sum_{j' \in [m]} \beta_{j',t+1} \cdot E_{j',x_{t-o+2},...,x_{t+1}} \cdot T_{j,j'}}_{\text{background state transitions}} + \underbrace{\sum_{l \in [k]} \beta_{j,t+|W_l|+1} \cdot \mathscr{L}\left(W_l; x_{t+1}, ..., x_{t+|W_l|}\right) \cdot E_{j,x_{t-o+|W_l|+2},...,x_{t+|W_l|+1}} \cdot G_{j,l}}_{\text{TF state transitions}}$$

Note that when $t > L - |W_l|$, some observable variables are missing for the full calculation of the TF state transition. In these locations, the contribution of those components to the summation is zero, meaning that our HOP-HMM avoids the transition into a TF state at a location where the PWM is too long to fit into the sequence X.

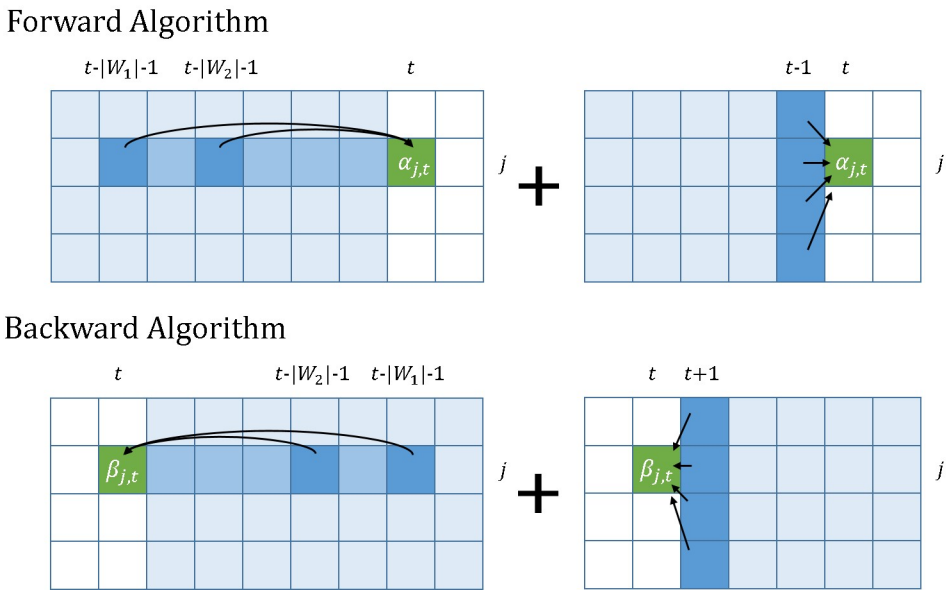


Figure 15: Forward-backward dynamic algorithm for HOP-HMM. In HOP-HMM, the tables α and β cells are filled from both the adjacent background states transitions and the background states preceding or proceeding the motifs emitted by the TF states.

We will now explain why the described calculations result in

$$\alpha_{j,t} = P(y_t = (j,0), x_{1:t}) \qquad \beta_{j,t} = P(x_{t+1:L} | y_t = (j,0), x_{1:t})$$

starting with the forward probabilities matrix α.

From the law of total probability, the probability $\alpha_{j,t}$ is the sum of probabilities of all the possible transitions that ended in the background state (j,0):

$$\alpha_{j,t} = P\left(y_t = (j,0), x_{1:t}\right) =$$
$$= \underbrace{\sum_{j' \in [m]} P\left(y_{t-1} = (j',0), y_t = (j,0), x_{1:t}\right)}_{\text{background state transitions}} + \underbrace{\sum_{l \in [k]} P\left(y_{t-|W_l|-1} = (j,0), y_{t-|W_l|:t-1} = (j,l), x_{1:t}\right)}_{\text{TF state transitions}}$$

The right-side term of a TF state transition can be split with the chain rule to:

$$P\left(y_{t-|W_l|:t-1} = (j,l), y_{t-|W_l|-1} = (j,0), x_{1:t}\right) =$$
$$P\left(y_{t-|W_l|-1} = (j,0), x_{1:t-|W_l|-1}\right) \cdot P\left(y_{t-|W_l|:t-1} = (j,l) | y_{t-|W_l|-1} = (j,0), x_{1:t-|W_l|-1}\right) \cdot$$
$$\cdot P\left(x_{t-|W_l|:t-1} | y_{t-|W_l|:t-1} = (j,l), y_{t-|W_l|-1} = (j,0), x_{1:t-|W_l|-1}\right) \cdot$$
$$\cdot P\left(x_t | y_t = (j,0), x_{1:t-1}, y_{t-|W_l|:t-1} = (j,l), y_{t-|W_l|-1} = (j,0)\right)$$

Since x_t is dependent only on y_t and $x_{t-o:t-1}$ and since y_t is dependent only on y_{t-1}, we can simplify the probabilities:

$$P\left(y_{t-|W_l|-1} = (j,0), x_{1:t-|W_l|-1}\right) \cdot P\left(y_{t-|W_l|:t-1} = (j,l) | y_{t-|W_l|-1} = (j,0)\right) \cdot$$
$$\cdot P\left(x_{t-|W_l|:t-1} | y_{t-|W_l|:t-1} = (j,l)\right) \cdot P\left(x_t | y_t = (j,0), x_{t-o:t-1}\right) =$$
$$= \alpha_{j,t-|W_l|-1} \cdot G_{j,l} \cdot \mathscr{L}\left(W_l; x_{t-|W_l|}, \ldots, x_{t-1}\right) \cdot E_{j, x_{t-o+1}, \ldots, x_t}$$

This process is similar for the left-side background state transitions. Using the chain rule:

$$P\left(y_{t-1} = (j',0), y_t = (j,0), x_{1:t}\right) =$$
$$= P\left(y_{t-1} = (j',0), x_{1:t-1}\right) \cdot P\left(y_t = (j,0) | y_{t-1} = (j',0), x_{1:t-1}\right) \cdot P\left(x_t | y_t = (j,0), y_{t-1} = (j',0), x_{1:t-1}\right) =$$
$$= P\left(y_{t-1} = (j',0), x_{1:t-1}\right) \cdot P\left(y_t = (j,0) | y_{t-1} = (j',0)\right) \cdot P\left(x_t | y_t = (j,0), x_{1:t-1}\right) =$$
$$= \alpha_{j',t-1} \cdot T_{j',j} \cdot E_{j, x_{t-o+1}, \ldots, x_t}$$

In order to explain the backward probabilities β, we will use the law of total probability:

$$\beta_{j,t} = P\left(x_{t+1:L} | y_t = (j,0), x_{1:t}\right) =$$
$$= \underbrace{\sum_{j' \in [m]} P\left(y_{t+1} = (j',0), x_{t+1:L} | y_t = (j,0), x_{1:t}\right)}_{\text{background state transitions}} +$$
$$+ \underbrace{\sum_{l \in [k]} P\left(y_{t+1:t+|W_l|} = (j,l), y_{t+|W_l|+1} = (j,0), x_{t+1:L} | y_t = (j,0), x_{1:t}\right)}_{\text{TF state transitions}}$$

For the background state transition term, we can use the chain rule and the Markovian independence of the transitions and emissions:

$$P\left(y_{t+1}=(j',0),x_{t+1:L}|y_t=(j,0),x_{1:t}\right)=$$
$$=P\left(x_{t+2:L}|y_{t+1}=(j',0),y_t=(j,0),x_{1:t+1}\right)\cdot P\left(x_{t+1}|y_{t+1}=(j',0),y_t=(j,0),x_{1:t}\right)\cdot$$
$$\cdot P\left(y_{t+1}=(j',0)|y_t=(j,0),x_{1:t}\right)=$$
$$=P\left(x_{t+2:L}|y_{t+1}=(j',0),x_{1:t+1}\right)\cdot P\left(x_{t+1}|y_{t+1}=(j',0),x_{1:t}\right)\cdot P\left(y_{t+1}=(j',0)|y_t=(j,0)\right)=$$
$$=\beta_{j',t+1}\cdot E_{j',x_{t-o+2},\ldots,x_{t+1}}\cdot T_{j,j'}$$

For the TF state transition term, we use once more the chain rule, followed by the simplification using the independencies of HOP-HMM:

$$P\left(y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0),x_{t+1:L}|y_t=(j,0),x_{1:t}\right)=$$
$$=P\left(x_{t+|W_l|+2:L}|y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0),y_t=(j,0),x_{1:t+|W_l|+1}\right)\cdot$$
$$\cdot P\left(x_{t+|W_l|+1}|y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0),y_t=(j,0),x_{1:t+|W_l|}\right)\cdot$$
$$\cdot P\left(x_{t+1:t+|W_l|}|y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0),y_t=(j,0),x_{1:t}\right)\cdot$$
$$\cdot P\left(y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0)|y_t=(j,0),x_{1:t}\right)=$$
$$=P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)\cdot P\left(x_{t+|W_l|+1}|y_{t+|W_l|+1}=(j,0),x_{t-o+|W_l|+1:t+|W_l|}\right)\cdot$$
$$\cdot P\left(x_{t+1:t+|W_l|}|y_{t+1:t+|W_l|}=(j,l)\right)\cdot P\left(y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0)|y_t=(j,0)\right)=$$
$$=\beta_{j,t+|W_l|+1}\cdot E_{j,x_{t-o+|W_l|+1},\ldots,x_{t+|W_l|+1}}\cdot\mathscr{L}\left(W_l;x_{t+1},\ldots,x_{t+|W_l|}\right)\cdot G_{j,l}$$

Using the forward and the backward probability matrices α and β, we can calculate two more probabilities that will help us to achieve the auxiliary probabilities (16), (17) and (18).

The first is the $m\times k\times L$ array ψ. We will use the chain rule several times in the first two steps, and in the third step we will simplify the 5 terms by the independencies of HOP-HMM:

$$\psi_{j,l,t}=P\left(y_t=(j,0),y_{t+1}=(j,l),x_{1:L}\right)=P\left(y_t=(j,0),y_{t+1}=(j,l),y_{t+|W_l|+1}=(j,0),x_{1:L}\right)=$$
$$=P\left(y_t=(j,0),y_{t+1}=(j,l),y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)\cdot P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)=$$
$$=P\left(x_{1:t},y_t=(j,0)\right)\cdot P\left(y_{t+1}=(j,l)|x_{1:t},y_t=(j,0)\right)\cdot P\left(x_{t+1:t+|W_l|}|y_{t+1:t+|W_l|}=(j,l),x_{1:t},y_t=(j,0)\right)\cdot$$
$$\cdot P\left(x_{t+|W_l|+1}|x_{t+|W_l|+2:L},y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)\cdot P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)=$$
$$=P\left(x_{1:t},y_t=(j,0)\right)\cdot P\left(y_{t+1}=(j,l)|y_t=(j,0)\right)\cdot P\left(x_{t+1:t+|W_l|}|y_{t+1:t+|W_l|}=(j,l)\right)\cdot$$
$$\cdot P\left(x_{t+|W_l|+1}|y_{t+|W_l|+1}=(j,0)\right)\cdot P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)=$$
$$=\alpha_{j,t}\cdot G_{j,l}\cdot L_{W_l,}\left(x_{t+1},\ldots,x_{t+|W_l|}\right)\cdot E_{j,x_{t+|W_l|-o+2},\ldots,x_{t+|W_l|+1}}\cdot\beta_{j,t+|W_l|+1}$$

The second probability is the likelihood of the observed sequence $x_{1:L}$. We will use the law of total probability followed by the chain rule:

$$P\left(x_{1:L}\right)=\sum_{j\in[m]}\left(P\left(y_t=(j,0),x_{1:L}\right)+\sum_{l\in[k]}P\left(y_t=(j,l),x_{1:L}\right)\right)=$$

$$= \sum_{j\in[m]} \left(P(x_{t+1:L}|y_t = (j,0), x_{1:t}) \cdot P(y_t = (j,0), x_{1:t}) + \sum_{l\in[k],\ t'\in[|W_l|]} P(y_{t-t'} = (j,0), y_{t-t'+1} = (j,l), x_{1:L}) \right) =$$

$$= \sum_{j\in[m]} \left(\alpha_{j,t} \cdot \beta_{j,t} + \sum_{l\in[k],\ t'\in[|W_l|]} \psi_{j,l,t-t'} \right)$$

We will now calculate the three auxiliary probabilities. First is the $m \times L$ array γ (16) which holds the probability distribution of the background state at a given position, given the sequence $x_{1:L}$:

$$\gamma_{j,t} = P(y_t = (j,0)|x_{1:L}) = \frac{P(y_t = (j,0), x_{1:t}) \cdot P(x_{t+1:L}|x_{1:t} y_t = (j,0))}{P(x_{1:L})} = \frac{\alpha_{j,t} \cdot \beta_{j,t}}{P(x_{1:L})}$$

Second is the $m \times m \times L$ array ξ (17) which holds the probability distribution of the background-to-background state transitions, given the sequence $x_{1:L}$:

$$\xi_{j_1,j_2,t} = P(y_{t-1} = (j_1,0), y_t = (j_2,0)|x_{1:L}) = \frac{P(y_{t-1} = (j_1,0), y_t = (j_2,0), x_{1:L})}{P(x_{1:L})} =$$

$$= \frac{P(x_{1:t-1}, y_{t-1} = (j_1,0), y_t = (j_2,0)) \cdot P(x_{t:L}|y_t = (j_2,0), x_{1:t-1})}{P(x_{1:L})} =$$

$$= \frac{P(x_{1:t-1}, y_{t-1} = (j_1,0)) \cdot P(y_t = (j_2,0)|y_{t-1} = (j_1,0))}{P(x_{1:L})} \cdot$$

$$\cdot \frac{P(x_t|y_t = (j_2,0), x_{1:t-1}) \cdot P(x_{t+1:L}|y_t = (j_2,0), x_{1:t-1})}{P(x_{1:L})} =$$

$$= \frac{\alpha_{j_1,t-1} \cdot T_{j_1,j_2} \cdot E_{j_2,x_{t-o+1},\ldots,x_t} \cdot \beta_{j_2,t}}{P(x_{1:L})}$$

Finally, the $m \times k \times L$ array η (18) which holds the probability distribution for background-to-background state transitions, given the sequence $x_{1:L}$:

$$\eta_{j,l,t} = P(y_{t-1} = (j,0), y_t = (j,l)|x_{1:L}) = \frac{\psi_{j,l,t}}{P(x_{1:L})}$$

Now, with (16), (17) and (18) at hand, we can complete the M-step and update θ by assigning the updates of (12), (13), (14) and (15).

The iterative execution of the adjusted E-step and M-step described in this section concludes the adaptation of the Baum-Welch algorithm for HOP-HMM:

Algorithm 5 Baum-Welch for HOP-HMM

Input:

X - Observed DNA sequence

Algorithm:

for s=[1...MAX_EM_ITERATIONS]:

 # E-step

$\alpha = \text{hop\_forward\_algorithm}(x_{1:L})$

$\beta = \text{hop\_backward\_algorithm}(x_{1:L})$

for $j = [1,...,m]$, $l = [1,...,k]$, $t = [1,...,L]$:

$$\psi_{j,l,t} = \begin{cases} \alpha_{j,t} \cdot G_{j,l} \cdot L_{W_l,}\left(x_{t+1:t+|W_l|}\right) \cdot E_{j,x_{t+|W_l|-o+2},...,x_{t+|W_l|+1}} \cdot \beta_{j,t+|W_l|+1} & |\, t+|W_l|+1 \leq L \\ 0 & |\, otherwise \end{cases}$$

$$Px = \sum_{j \in [m]} \alpha_{j,L}$$

for $j = [1,...,m]$, $t = [1,...,L]$:

$$\gamma_{j,t} = \frac{\alpha_{j,t} \cdot \beta_{j,t}}{Px}$$

for $j = [1,...,m]$, $l = [1,...,k]$, $t = [1,...,L]$:

$$\eta_{j,l,t} = \frac{\psi_{j,l,t}}{Px}$$

for $j_1 = [1,...,m]$, $j_2 = [1,...,m]$, $t = [1,...,L]$:

$$\xi_{j_1,j_2,t} = \frac{\alpha_{j_1,t-1} \cdot T_{j_1,j_2} \cdot E_{j_2,x_{t-o+1},...,x_t} \cdot \beta_{j_2,t}}{Px}$$

 # M-step

for $j = [1,...,m]$:

$$\pi_j = \gamma_{j,1}$$

for $b_1,...,b_o = [1,...,1]$,...,$[4,...,4]$:

$$E_{j,b_1,b_2,...,b_o} = \frac{\sum_{t \in o,...,L} \gamma_{j,t} \cdot 1_{b_1,...,b_o}(x_{t-o+1},...,x_t)}{\sum_{t \in o,...,L} \gamma_{j,t}}$$

for $l = [1,...,k]$:

$$G_{j,l} = \frac{\sum\limits_{t \in 2,...,L} \eta_{j,l,t}}{\sum\limits_{t \in 1,...,L-1} \gamma_{j_1,t}}$$

for $j_2 = [1,...,m]$:

$$T_{j,j_2} = \frac{\sum\limits_{t \in 2,...,L} \xi_{j,j_2,t}}{\sum\limits_{t \in 1,...,L-1} \gamma_{j,t}}$$

If θ converged, break EM for loop

 # EM converged or stopped

return θ

The Baum-Welch algorithm is described with the input of a single sequence of observable variables $x_{1:L}$. In reality, we are often dealing with the task of learning $\hat{\theta}$ from multiple sequences at once. In HOP-HMM we use the multi-sequence method as Rabiner (1989), where the E-step probabilities are calculated separately for each sequence, and in the M-step where positions from all sequences are summed for the parameters update.

2.3 Hidden Sequence Inference

Acquiring the maximal likelihood estimation $\hat{\theta}$ from the Baum-Welch algorithm opens the door to several needed inferences given a sequence $x_{1:L}$:

1. Most likely hidden state at any position in a sequence
2. Most likely hidden sequence
3. Dominant hidden state in a short sequence

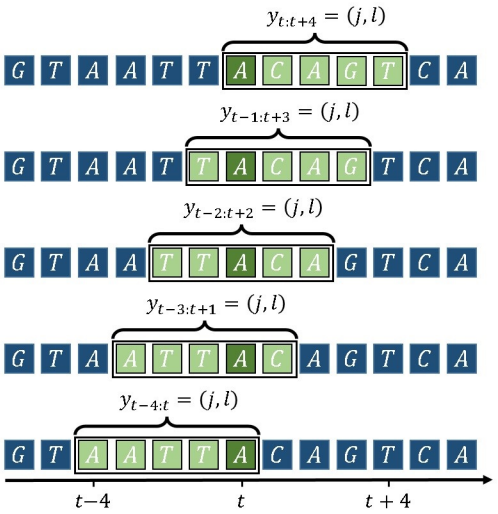
γ (16) and η (18) can be used to solve inference 1 for HOP-HMM. Here we aim to maximize here a posterior probability in a specific position:

$$y_t^* = \underset{j\in[m],l\in[k]\cup\{0\}}{argmax} P(y_t = (j,l)|x_{1:L})$$

In a regular HMM, we could approximate this by taking the state with the maximal posterior probability from γ (10) built by a $\hat{\theta}$ that was obtained by the Baum-Welch algorithm. In HOP-HMM, γ (16) is not sufficient since it only holds the probability of being in background state $P\left(y_t = (j,0)|x_{1:L};\hat{\theta}\right)$. In order to calculate the posterior probability $P\left(y_t = (j,l)|x_{1:L};\hat{\theta}\right)$ for TF states, where $l > 0$, we sum all options of a TF state (j,l) that covers position t as described in (16).

$$P\left(y_t = (j,l)|x_{1:L};\hat{\theta}\right) = \sum_{i\in[|W_l|]} P\left(y_{t-i+1:t-i+|W_l|} = (j,l)|x_{1:L};\hat{\theta}\right) =$$

$$= \sum_{i\in[|W_l|]} P\left(y_{t-i} = (j,0), y_{t-i+1} = (j,l)|x_{1:L};\hat{\theta}\right) = \sum_{i\in[|W_l|]} \eta_{t-i+1,j,l}$$



$$P(y_t = (j,l)|x_{1:L}) = \sum_{i=1}^{|W_l|} P\left(y_{t-i+1:t+|W_l|-i} = (j,l)\middle|x_{1:L}\right)$$

Figure 16: $P(y_t = (j,t)|x_{1:L})$ is the posterior probability to be in TF state (j,l) at position t, marked in dark green. It is equal to the sum of probabilities of entering into the TF state before position t. In this example, W_l is a PWM of length 5, therefore it sums 5 different possible positions that include y_t, marked in light green.

Choosing the maximum value over $P\left(y_t = (j,l)|x_{1:L};\hat{\theta}\right)$ and $P\left(y_t = (j,0)|x_{1:L};\hat{\theta}\right)$ will give us the most likely state of $\hat{y}_t$:

$$\hat{y}_t = \underset{j\in[m],l\in[k]\cup\{0\}}{argmax} \gamma_{j,t} \cup \sum_{i\in[|W_l|]} \eta_{t-i+1,j,l} \tag{19}$$

Inference 2 aims to reach the most likely hidden sequence:

$$y^*_{1:L} = argmax_{y_{1:L}} P\left(y_{1:L}|x_{1:L};\hat{\theta}\right) \tag{20}$$

The main dissimilarity with inference 1 is the consideration of the dependency between adjacent states. In inference 1, for example, two adjacent positions may be individually inferred states between which the transition probability equals 0. In such a case, even though each state maximizes the posterior probability at its own position, the likelihood of the resulting sequence would be 0 since it contains an impossible transition. As a consequence, the requirement in inference 2 for the most likely hidden sequence, with the transitions inside it taken into account, could not simply be achieved by concatenating all the most likely states at every position into a sequence.

2.4 Viterbi Algorithm Adaptation

In HMM, deriving the maximal likelihood hidden sequence of (20) is done by the Viterbi algorithm, named after Andrew Viterbi who proposed it in Viterbi (1967). The Viterbi algorithm resembles the forward algorithm, with two main differences:

1. Maximization replaces summation over the possible transitions.
2. Indices of the states with the maximal likelihood are kept in the filling of V^2, and are eventually used to backtrack the chosen states in the most likely path.

Algorithm 6 Viterbi Algorithm

Input:

θ- HMM parameters $\{\pi, T, E\}$

$x_{1:L}$ - Observed DNA sequence

Algorithm:

$for\ j = [1,...,m]:$

$V^1_{j,1} = \pi_j \cdot E_{j,x_1}$

$V^2_{j,1} = 0$

$for\ t = [2,...,L]:$

$for\ j = [1,...,m]:$

$V^1_{j,t} = max_{j'\in[m]}\left(V_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_t}\right)$

$V^2_{j,t} = argmax_{j'\in[m]}\left(V_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_t}\right)$

 # back tracing

$\hat{y}_L = argmax_j V^1_{j,L}$

for $t = [L,...,2]$:

$\hat{y}_{t-1} = V^2_{y_t,t}$

return $\hat{y}_{1:L}$

For HOP-HMM, the Viterbi algorithm is adapted into to HOP-HMM in two ways:

- Maximization is done over two types of state transition probabilities: background-to-background and background-to-TF, held in A and B vectors.
- The backtracking indices held in V^2 tables are two indices, since states in HOP-HMM are described by two indices.

Algorithm 7 Viterbi Algorithm for HOP-HMM

Input:

θ- HOP-HMM parameters $\{\pi, T, G, E\}$

$x_{1:L}$ - Observed DNA sequence

Algorithm:

$for\ j = [1,...,m]:$

$V^1_{j,1} = \pi_j \cdot E_{j,x_1}$

$V^2_{j,1} = 0$

$for\ t = [2,...,L]:$

$for\ j = [1,...,m]:$

 # background-to-background state transition

$A = \left\{V_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_{t-o+1},...,x_t} | j' \in [m]\right\}$

 # background-to-TF state state transition

$B = \left\{V_{j,t-|W_l|-1} \cdot G_{j,l} \cdot \mathscr{L}\left(W_l; x_{t-|W_l|},...,x_{t-1}\right) \cdot E_{j,x_{t-o+1},...,x_t} | l \in [k]\right\}$

$V^1_{j,t} = max\left(A \cup B\right)$

$$V^2_{j,t} = \begin{cases} (argmax(A), 0) & max(A) > max(B) \\ (j, argmax(B)) & otherwise \end{cases}$$

$\hat{y}_L = \left(argmax_j V^1_{j,L}, 0\right)$ *# mandatory background state at the end of the sequence*

$t = L$

while $t > 1$:

$(j,l) = V^2_{y_t[0],t}$

if $l = 0$: *# if* $l = 0$ *the hidden state at* $t-1$ *is a background state*

$\hat{y}_{t-1} = (j, 0)$

$t = t - 1$

else:

$\hat{y}_{t-|W_l|:t-1} = (j,l)$

$\hat{y}_{t-|W_l|-1} = \hat{y}_t$

$t = t - |W_l| - 1$

return $\hat{y}_{1:L}$

The Viterbi algorithm outputs a sequence of hidden states, also called a Viterbi path, which can be used to evaluate the trained model by comparing it to the epigenetic data, as done in this work. In cases where the exact true boarders of the active element are unknown due to noisy data, short DNA sequences can be classified by their dominant states. We found this method useful in our preliminary evaluation of the algorithm, but did not include it in the results of this work. This simplistic classification is made by choosing the most abundant state in the estimated Viterbi path $\hat{y}_{1:L}$:

$$y_{class} = mode_{t \in [L]} \hat{y}_t$$

3 Results

3.1 Synthetic Data

In order to evaluate HOP-HMM, we first measured its capabilities on synthetic DNA dataset that was created in a controlled way. We could then experiment with HOP-HMM on real human DNA sequences. The evaluation process on synthetic data was performed through the following steps (see figure 17):

1. We generated parameters for a HOP-HMM θ, which were treated as the true parameters. The generation process of θ is done in the following way:

   - Each cell in T was sampled from a uniform distribution

   $$T_{i,j} \sim U\left(minT_{i,j}, maxT_{i,j}\right) \tag{21}$$

   - Each cell in G was sampled both from a uniform and from a Bernoulli distribution

   $$G_{i,j} \sim U\left(minG, noiseG \cdot maxG\right) + 1_{(i,0)\in ENH} \cdot Bern\left(\frac{ActiveTFs}{k}\right) \cdot maxG \tag{22}$$

   where

   $$1_{(i,0)\in ENH} = \begin{cases} 1 & (i,0) \in ENH \\ 0 & otherwise \end{cases}$$

   ENH is the set of "enhancer-mimicking" background states, which are predefined background states that have a high probability of transitioning into TF states. The rest of the background states will have a low probability to create TFBS, since we want some of the states to model sparse TFBSs (non-regulatory elements) surrounding the enhancers. In our experiments ENH contained all but one state: $(m,0)$ meaning that one background state had almost no TFBS and the other $m-1$ background states did have TFBSs.

   - After being sampled, T and G cells were divided element-wise by the sum of their rows, so that the ensemble of every row of T and its corresponding row of G became a distribution:

   $$T_{i,j} = \frac{T_{i,j}}{\sum_{j'\in[m]} T_{i,j'} + \sum_{j'\in[k]} G_{i,j'}} \qquad G_{i,j} = \frac{G_{i,j}}{\sum_{j'\in[m]} T_{i,j'} + \sum_{j'\in[k]} G_{i,j'}} \tag{23}$$

   - E was sampled from a uniform distribution $E_{j,b_1,\ldots,b_o} \sim U(0,1)$ and divided by the sum of its last index to become a distribution array, as in the previous step:

   $$E_{j,b_1,\ldots,b_o} = \frac{E_{j,b_1,\ldots,b_o}}{\sum_{b'=[4]} E_{j,b_1,\ldots,b_{o-1},b'}}$$

   - The start state distribution π was non-random, and was set so that the first states were always one of the non-enhancer background states

   $$\pi_i = \frac{1_{(i,0)\notin ENH}}{m - |ENH|}$$

2. Sequences were generated using the HOP-HMM with the true θ. Both the observed and the hidden sequences were used, denoted X and Y. We split the X and Y sequences into train and test sections for cross validation.

3. We trained a $\hat{\theta}$ on the DNA sequences of X_{train} with the Baum-Welch algorithm for HOP-HMM.

4. With the trained parameters $\hat{\theta}$, we estimated $\hat{Y}_{test}$ from X_{test} and $\hat{Y}_{train}$ from X_{train} using the Viterbi algorithm adaptation to HOP-HMM. We also calculated the posterior probability of $P\left(y_t|x_{1:L};\hat{\theta}\right)$ from X_{test} and X_{train} using the forward-backward algorithm for HOP-HMM. These results were then compared to the real Y_{test} and Y_{train} to check for accuracy.

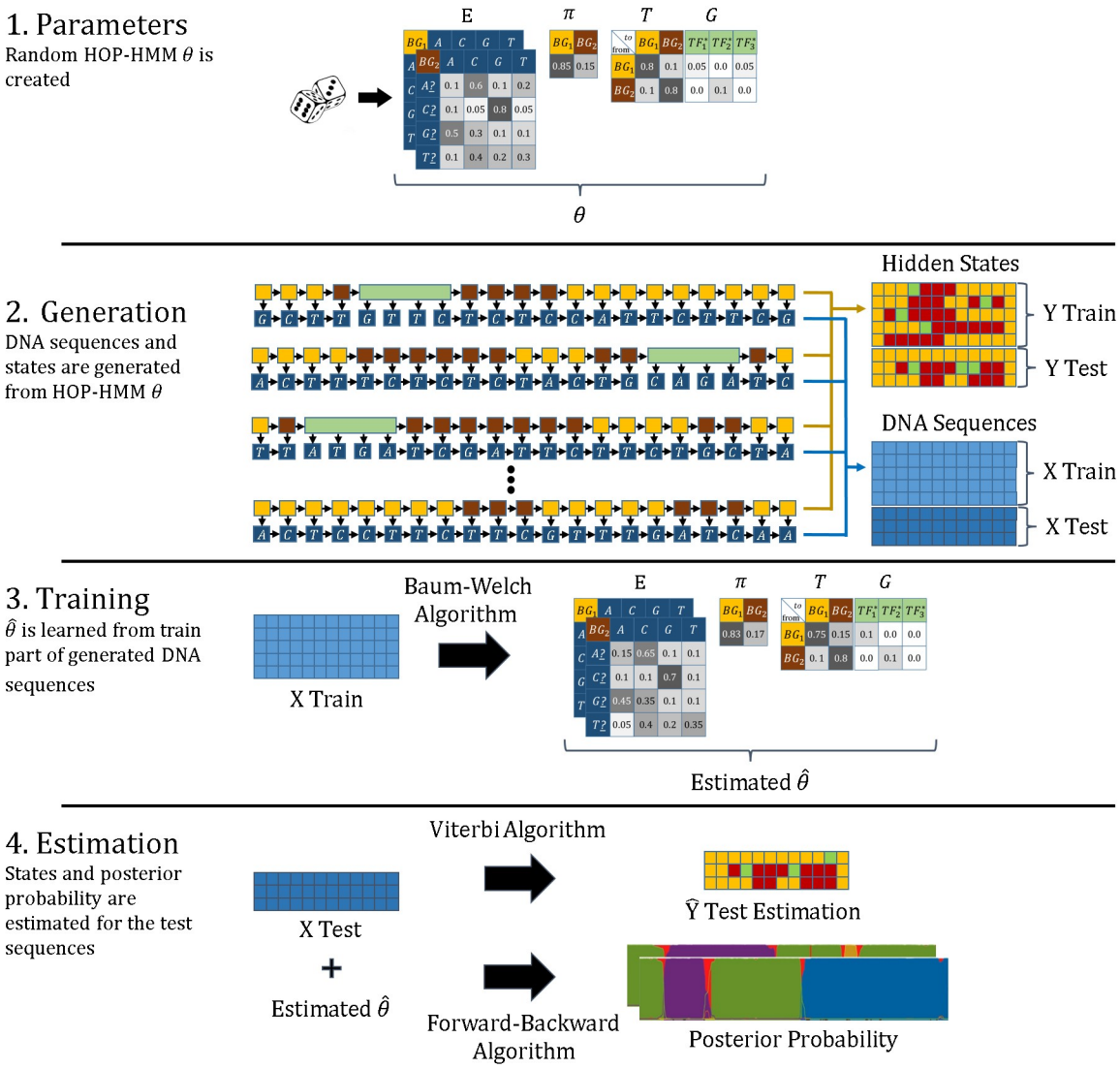


Figure 17: Workflow of the evaluation process. A θ is sampled and a HOP-HMM is created, with which several fixed-length sequences are generated. A new model $\hat{\theta}$ is then fitted to the train section of the observed sequences, via a Baum-Welch algorithm. With $\hat{\theta}$, a hidden sequence is then estimated by a Viterbi algorithm adapted for HOP-HMM, and a posterior probability estimation is calculated by (19).

In the synthetic data evaluation we used the following values for the hyperparameters:

Constant Name	Values
L	1500
N_{train}, N_{test}	425, 75
m	5
k	25
o	3
$minG$	10^{-7}
$maxG$	10^{-3}
$noiseG$	$5 \cdot 10^{-2}$
$ActiveTFs$	3
$minT$	$\begin{pmatrix} 1-10^{-2} & 10^{-7} & 10^{-7} & 10^{-7} & 10^{-3} \\ 10^{-7} & 1-10^{-2} & 10^{-7} & 10^{-7} & 10^{-3} \\ 10^{-7} & 10^{-7} & 1-10^{-2} & 10^{-7} & 10^{-3} \\ 10^{-7} & 10^{-7} & 10^{-7} & 1-10^{-2} & 10^{-3} \\ 10^{-5} & 10^{-5} & 10^{-5} & 10^{-5} & 1-10^{-4} \end{pmatrix}$
$maxT$	$\begin{pmatrix} 1-10^{-3} & 10^{-5} & 10^{-5} & 10^{-5} & 5 \cdot 10^{-3} \\ 10^{-5} & 1-10^{-3} & 10^{-5} & 10^{-5} & 5 \cdot 10^{-3} \\ 10^{-5} & 10^{-5} & 1-10^{-3} & 10^{-5} & 5 \cdot 10^{-3} \\ 10^{-5} & 10^{-5} & 10^{-5} & 1-10^{-3} & 5 \cdot 10^{-3} \\ 10^{-4} & 10^{-4} & 10^{-4} & 10^{-4} & 1-10^{-3} \end{pmatrix}$

The Baum-Welch algorithm ensures the increase of the likelihood for each step. However it does not ensure convergence to the optimal θ^* (Rabiner, 1989; Yang et al., 2015) since there is no known analytical way to reach it. Consequently, the Baum-Welch algorithm converges into a local maximum $\hat{\theta}$ that could be a relatively low likelihood estimation, depending on the initialization point of the first θ. During the initial evaluation of the inference EM algorithm, we noticed that many of the executions converged to local maxima which tended to overshoot the inter-states transition probability, resulting in a tendency to irregular Viterbi paths with frequent state changes.

We therefore addressed this issue in two ways that had a significant positive impact on the convergence rate and solution quality:

1. We used regularization for faster and better $\hat{\theta}$ convergence (see figures 19 and 20). Following the M-step updates of the first 5 EM iterations, we fixed the background states transition probabilities T to remain between $maxT$ and $minT$ matrices from (21):
   - If $T_{i,j} < minT_{i,j}$ then we set $T_{i,j} = minT_{i,j}$
   - If $T_{i,j} > maxT_{i,j}$ then we set $T_{i,j} = maxT_{i,j}$
   - T and G cells were divided by the sum of their rows, so that the ensemble of their rows remained a distribution as in (23)

2. Since Baum-Welch seeks local maxima, running it multiple times with different initializations would cause convergence for different $\hat{\theta}$ results. As could be expected, we observed throughout multiple initializations that the higher the log likelihood of final $\hat{\theta}$, the lower its root mean square error (RMSE) compared to the true θ (see figure 18). The RMSE measurement between θ and $\hat{\theta}$ required matching similar states by their distributions in E and G, which was done with the use of the Hungarian algorithm (Kuhn, 1955). This correlation is important since on observed real sequences only the estimated $\hat{\theta}$ likelihood is known, while the true θ is unknown. Therefore, in order to obtain an estimated $\hat{\theta}$ as close as possible to the true θ, one should redo several EM executions and choose the $\hat{\theta}$ with the highest log likelihood.

The experiment was performed on a dataset of 500 synthetic sequences (85% train, 15% test), all of them 1500 bp-long. The trained model had 5 hidden background states with an emission order of 3, and each background state had 25 TF states. The background states mean test precision was 98.5% and the mean test recall was 97.6%.

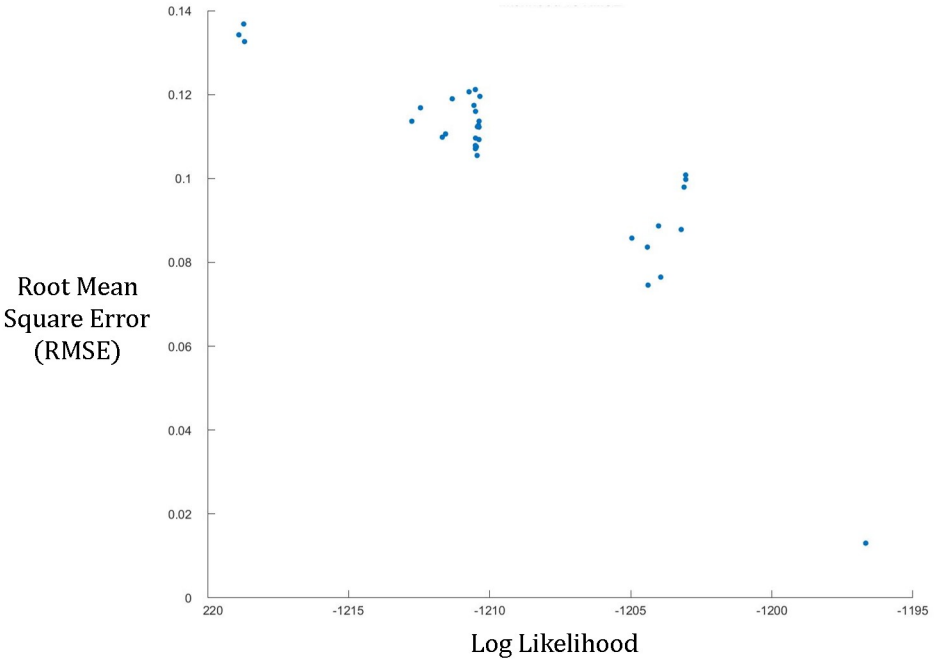


Figure 18: The log likelihood of $\hat{\theta}$ is correlated to its error. Over multiple runs of the Baum-Welch algorithm for HOP-HMM, $\hat{\theta}$ estimations with higher mean log likelihoods also had lower RMSE compared to the true θ.

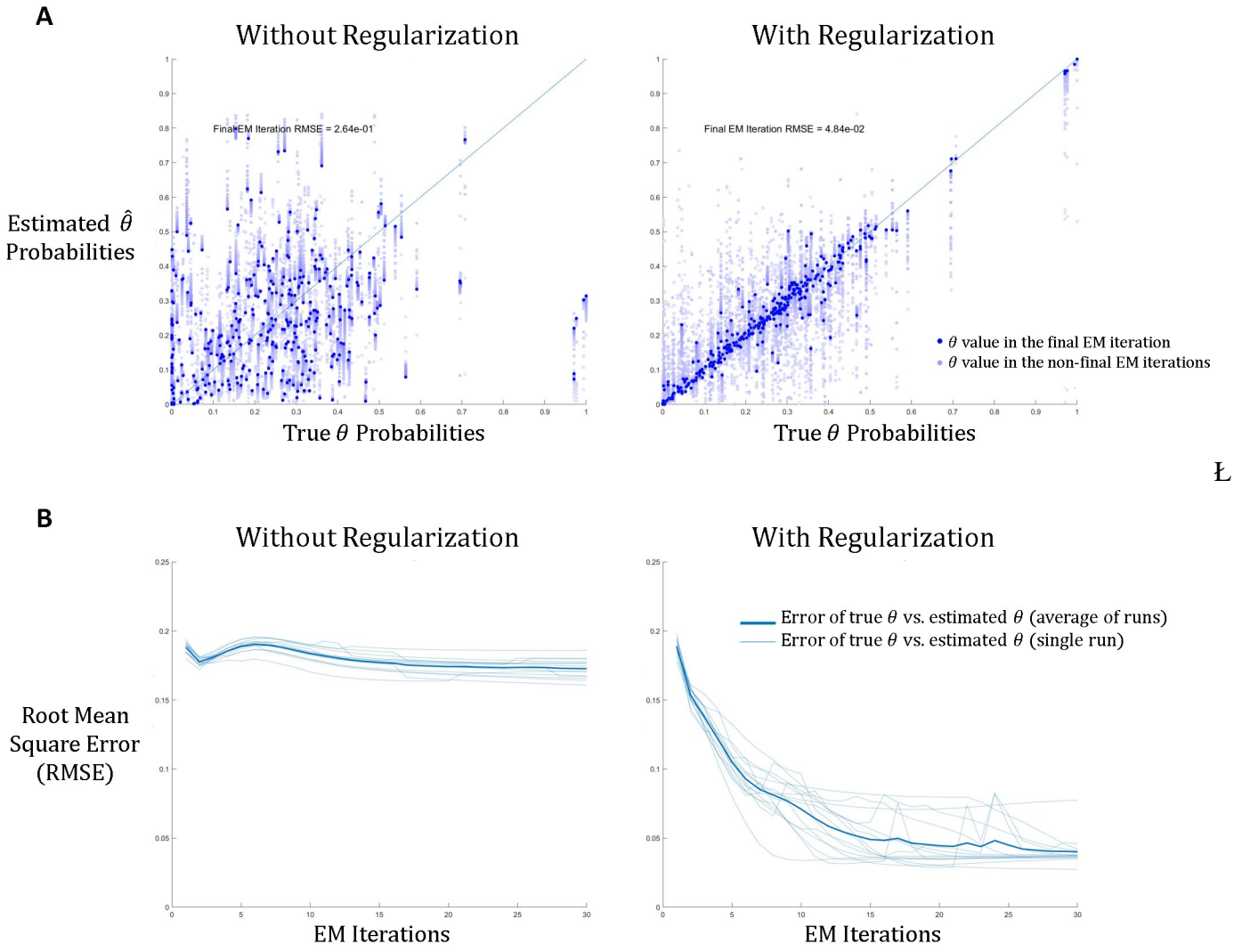


Figure 19: Regularization aids the EM to converge faster and with less error compared to the true parameters θ. **A)** The iterations of a single EM execution. The dots represent the updates of the $\hat{\theta}$ values during the EM iterations. The diagram shows that with the regularization enabled, the estimated $\hat{\theta}$ values mostly advance toward the true value of θ. **B)** The RMSE between true θ and estimated $\hat{\theta}$ during multiple EM executions. The regularization results in a significant error decrease during the EM iterations.

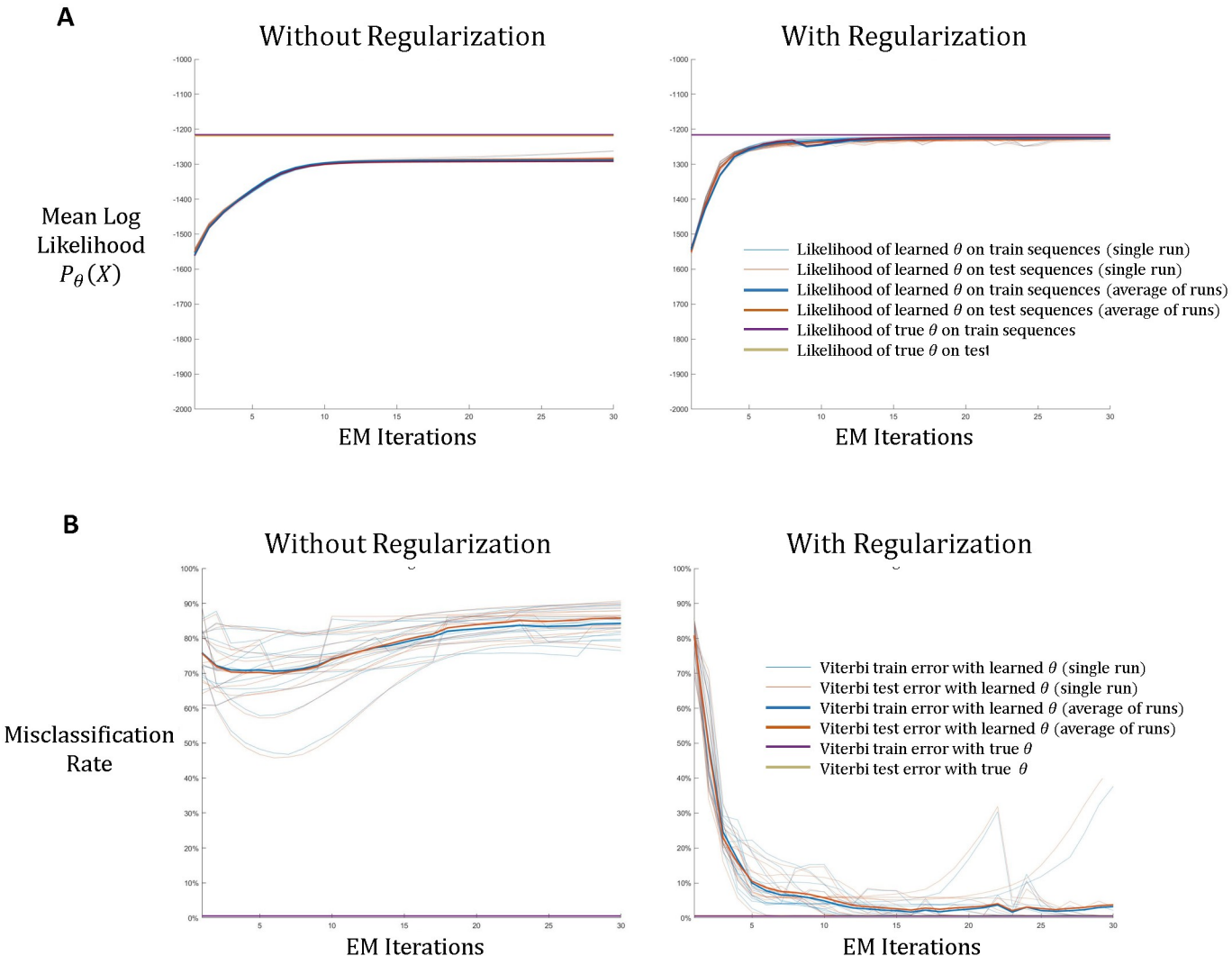


Figure 20: On average, the regularization causes to the EM algorithm to converge to $\hat{\theta}$ estimations with higher likelihoods and that produce Viterbi paths with lower misclassification rates. **A)** The mean log likelihood of the sequences increases until convergence during the EM iterations. With the regularization enabled, the likelihood is not always monotonous, since with it the likelihood increase of Baum-Welch updates is no longer guaranteed. **B)** During EM iterations, the learned $\hat{\theta}$ values yield a more accurate Viterbi path. Note that not even the true θ could produce a Viterbi path that perfectly matches the true hidden sequence, as the value of the purple and yellow charts is 0.1% and not 0%.

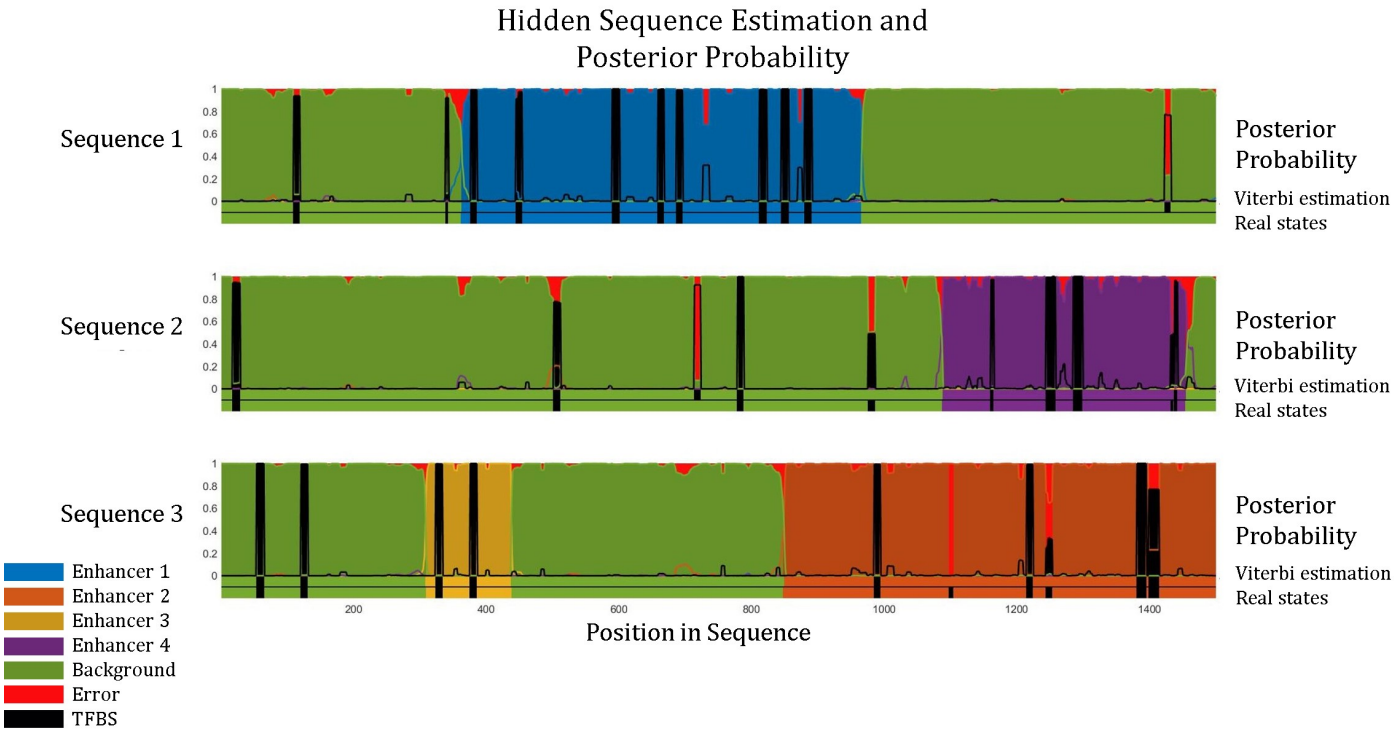


Figure 21: Posterior probability of sequences, estimated by a trained HOP-HMM $\hat{\theta}$ on a test dataset of sequences that were synthetically generated by a HOP-HMM θ. The Viterbi path by $\hat{\theta}$ and the true hidden states of each sequence are shown at the bottom of each posterior probability. The black TFBS is the sum of all the probabilities of being in any of the TF states.

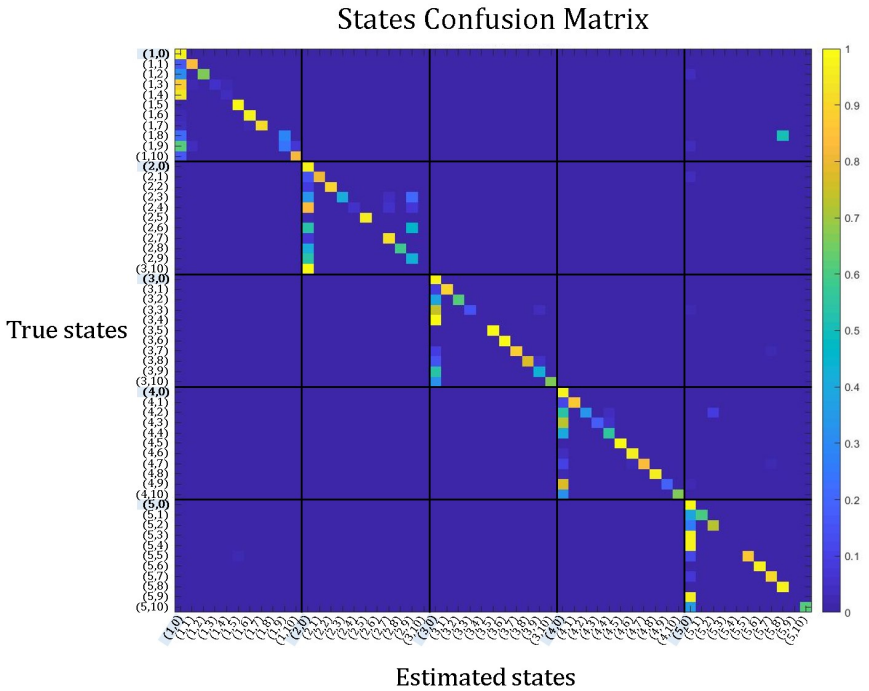


Figure 22: Confusion matrix of true and estimated states by the Viterbi algorithm for synthetic sequences. Rows are normalized so their sum is equal to 1. For a smaller confusion matrix that would be easier to present, the HOP-HMMs in this experiment contained 5 background states and only 10 TF states. Similarly to the experiment with the larger HOP-HMM shown in figure 21, the majority of true and estimated states are background states, with the highlighted indices $(1,0)$, $(2,0)$, $(3,0)$, $(4,0)$ and $(5,0)$. Note that the Viterbi algorithm often misclassifies TF states as their background state.

3.2 Human DNA Experiment

For the testing of HOP-HMM on human genetic data, we sought to assess if HOP-HMM could detect enhancers activity in two human tissues. Thus, we created a dataset of enhancer sequences based on epigenetic data collected from 57 tissues by the Roadmap project. In order to choose the location of the enhancer elements, we manipulated the Roadmap project BED files with BEDTools (Quinlan and Hall, 2010). We chose the intersection of DNase-I, H3K27ac and H3K4me1 peaks, while avoiding peaks of H3K27me3 and H3K4me3, and all sequences within 5000 bp from known genes. The sequences chosen were 5000 bp-long sequences from the hg19 assembly, centered around their DNase-I peak to ensure the flanks of the enhancer. Among these enhancers, we chose only sequences of tissue-specific enhancers in one of two types of tissues, which were selected out of these same 57 tissues. After some trial and error, we chose the somewhat arbitrary cutoff of the top 40% strongest DNase-seq peaks, which yielded enough sequences (around 500 sequences per tissue sample on average) with distinguishable distributions between the tissues. We added sequences with no known role from random locations in the genome, distant from genes or enhancers background sequences. A HOP-HMM was trained by the Baum-Welch algorithm on the collected sequences. The trained model was then used to produce a Viterbi estimated hidden states sequence and posterior probability, which could be compared to the epigenetic tracks.

For the set of PWMs used by the TF states of the HOP-HMM, we used a JASPAR dataset of 519 vertebrates PWMs, out of which we selected 50 PWMs for a practical run-time. The selected PWMs were chosen by three methods, each method being responsible for one third of these 50 PWMs:

- PWMs of TFs relatively expressed for one tissue compared to the other, according to the Roadmap RNA-seq data. This method does not depend on the sequences themselves, but on the epigenetic properties of the tissues.
- PWMs which were abundant in the sequences, i.e. PWMs with the highest mean likelihood of binding to sequences. The average likelihood of PWM W to bind to a sequence x was simplified as the mean of the three highest binding likelihoods in the sequence, as described in figure 3. Note that in order to compare between PWM likelihoods we used the PSSM form as in (1) for its length-independence property.
- PWMs that had stronger presence in sequences from one tissue when compared to the other. Specifically, the PWMs with sequence binding likelihoods (as defined in the previous method) that could best distinguish between the sequences from one tissue and the sequences of the other tissues in terms of AUC-ROC.

In our experiment, the posterior probability of some sequences had a good resemblance to the DNase-seq track, causing a good overlap between the Viterbi-path and the ChromHMM classifications (see figure 23), though such similarity did not always occur.

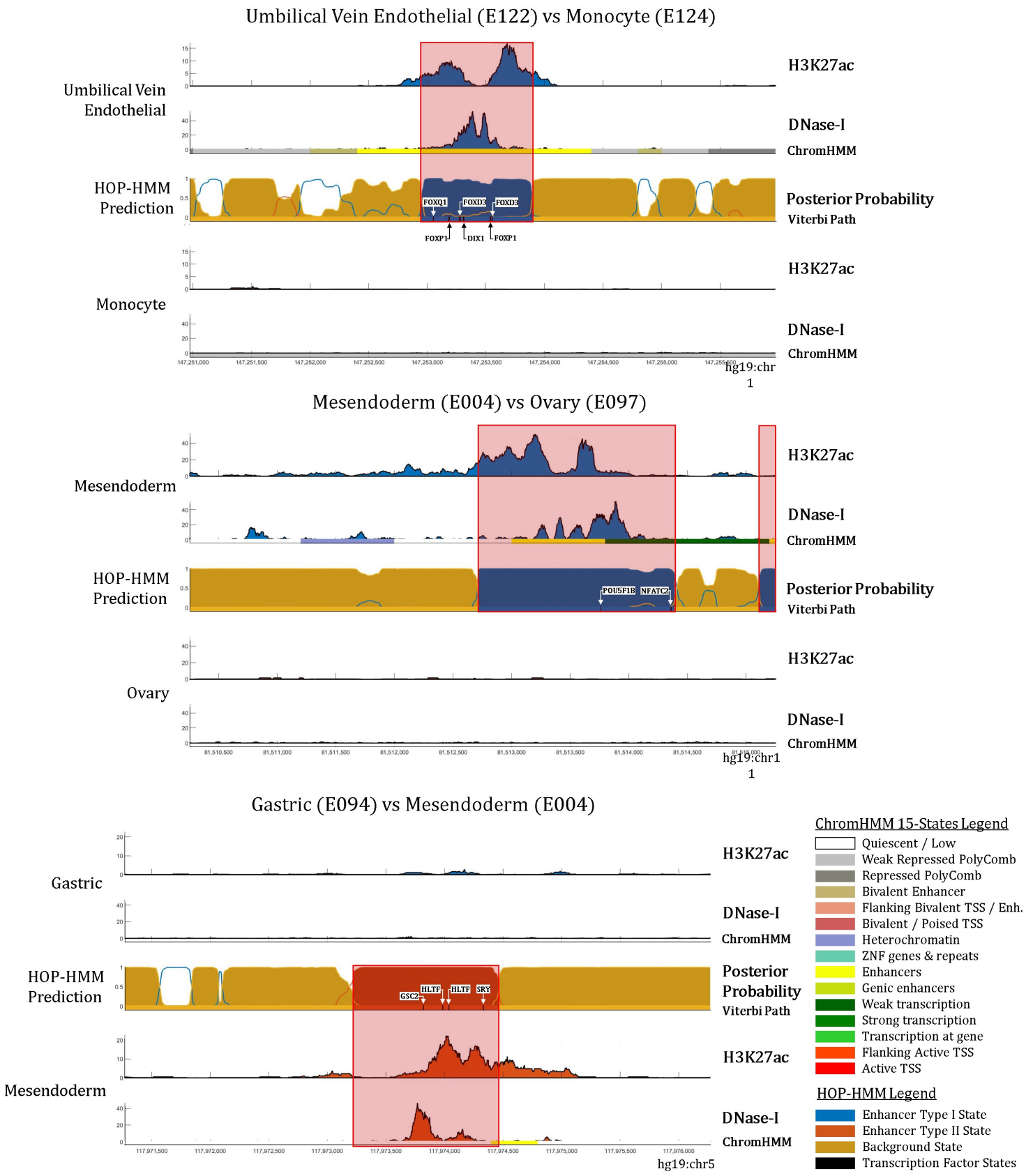


Figure 23: Examples of HOP-HMM classification of 5000 pb-long tissue specific enhancer sequences from the human genome. Each of the three graphs is the output of a different HOP-HMM with three background states (two enhancer states and one non-enhancer state) and 50 TF states, and each was trained on enhancers from the two chosen tissues. H3K27ac and DNase-I measurements are in $-log_{10}(p-value)$ units.

4 Discussion and Conclusions

In this work we aimed to develop a generalized HMM, HOP-HMM, tailored for the enhancer structure. We developed the mathematical adjustments to the different parts of the EM algorithm and provided reasoning for the correctness of inferring the altered model from the data. We also implemented a model and an algorithm in Matlab code for the evaluation on real or synthetic data, as described in the results. During the algorithm implementation, we overcame a few difficulties originating from the scale of the data, such as caching the costly response of the PWMs to the sequences during the forward-backward algorithms, and splitting sequences into batches in order to hold and manipulate the large η (18) array in the memory. The implementation also included a code for generating DNA sequences from a randomly selected HOP-HMM to which a different model could be fitted and compared. Naturally, the larger the generated dataset used to fit the model in the synthetic data experiment, the better the performance of the fitting. Overall, the generated DNA sequences experiment results were positive and provided evidence for the ability of the algorithm to train successfully on DNA sequences created under our HOP-HMM assumptions.

Like in other machine learning challenges, comparison to the state-of-the-art of prediction models for regulatory sequences is not straight forward. Several softwares accurately predict epigenetic features of DNA sequences: gkm-SVM by De Beer et al. (2014), DeepBind by Alipanahi et al. (2015), DeepSEA by Zhou and Troyanskaya (2015) and Basset by Kelley et al. (2016). However, each of them approaches the task with a different definition of the problem, and different datasets for it. Among these softwares, DeepSEA seems the most comprehensive since it was trained on the entire human genome and predicts the largest number of epigenetic features for multiple cell types. Though all these softwares extract substantial information about how sequences determine their epigenetic, they are all based on a model that defines the problem as a supervised learning task. Using a model that would define it as an unsupervised learning task (such as ours) holds potential for the *de novo* annotation of the entire genome, without the need for epigenetic data from extracted cells.

The implementation of the HOP-HMM for this work was done in Matlab, and its run-time and memory requirements for the training of significant parts of the human genome are still unreachable (several days for each EM iteration). In order to assess the ability of the model to detect real human enhancers on a smaller subset of the genome, we created a dataset of tissue-specific enhancers from two tissues and non-regulatory "background" sequences whose locations were deducted from the epigenetic data of the Roadmap project. Though some of these sequences were correctly classified by the trained model, no tested pair of tissues had a classification with consistent similarity to its epigenetic data, and our results certainly did not surpass those of DeepSEA. We found that some of the tissues had many well-classified enhancers, though no good classifications could be found for the rest. Over all, the small scale of the experiment we executed is not sufficient to determine whether the model could accurately predict enhancer sequences from the entire human genome. The causes of the mixed results of the human genome experiment compared to the positive results of the generated synthetic data experiment may reside in one of two reasons: either the EM algorithm did not converge toward good enough parameters, or no such parameters existed in the HOP-HMM hypothesis space.

As for the former possibility, the experiments done with our generated data showed a tradeoff between the amount of data provided to the EM algorithm, and its ability to converge toward parameters that could detect the difference between background states holding similar emission distributions (E and G) in the generating HOP-HMM parameters. In other words, convergence of the EM to low quality local maxima mainly occurred when the emission distributions in the generating parameters were not significantly different. This tradeoff is common in the machine learning field, since the ability of many models to distinguish between similar classes depends on the amount of relevant samples in the training dataset

(Dupin et al., 2011). In our dataset of human enhancers, most tissues only had several hundred enhancer sequences, which may have been insufficient for a high quality convergence of the EM algorithm.

As for the latter possibility, it could mean that the spanned solution space of the HOP-HMM assumption does not match our human enhancer dataset. This could stem from several reasons: wrong PWMs selection as hyperparameters, too small or noisy dataset building from the Roadmap data, and even a more complex enhancer structure than assumed by the HOP-HMM hypothesis.

In further research, the use of updated data with cleaner experiments and/or more tissue diversity would be likely to provide better results. Even without new data, a more efficient implementation of the algorithms would allow executing it on more significant parts of the genome within a reasonable time, which might result in better classifications. Improvements to HOP-HMM which should be tried in the future include the introduction of learning to the PWMs before or during EM iterations, or the entire replacement of the PWMs emissions by a different TFBS modeling method.

5 Appendix: Source Code

The code for this research was written in Matlab, and can be found at `https://github.com/David-Taub/HOP-HMM`.

Variable	Meaning
`L`	DNA sequences length
`N`	Number of DNA sequences
`m`	Number of background states
`k`	Number of TF states of each background state
`order`	Dependency order of the emission of the background states done by E. For example, if `order` equals 3, then the emission is conditional on 2 previous observable variables.
`backgroundAmount`	Number of background states which are non-enhancers by having low transition probability into TF states

- **HOP-HMM/data/peaks/scripts/download_and_process_all.sh**

  Linux bash script which downloads data files of epigenetic from Roadmap website, JASPAR PWMs and hg19 genome. After downloading, the data is per-processed with BEDtools and bigWigToBEDGraph. The only part in this project that requires Linux is the bigWigToBEDGraph.

- **HOP-HMM/src/+peaks/minimizeMergePeak.m**

  Reads downloaded bed files, processes them and saves them into MAT-file v7.3.

  ```
  params = genParams(m, k, backgroundAmount, L, order, doESharing, doGTBound);
  mergedPeaksMin = minimizeMergePeak(params, L);
  ```

  where `doGTBound` indicates whether or not to apply regularization on T and G transition probabilities and `doESharing` indicates whether or not to force E to share the emission across all background states

- **HOP-HMM/src/misc/genSyntheticMergedPeaksMin.m**

  Generates DNA sequences X and hidden variables Y out of a random θ, which was sampled by genTheta.m

  ```
  params = genParams(m, k, backgroundAmount, L, order, doESharing, doGTBound);
  mergedPeaksMin = genSyntheticMergedPeaksMin(N, L, params,
  startWithBackground, backgroundGNoise);
  ```

  where `startWithBackground` indicates whether or not to force π to allow starting only from non-enhancer background states and `backgroundGNoise` is the background rate of background-to-TF state transition, marked as *noiseG* in (22)

- **HOP-HMM/src/misc/genTheta.m**

  Generates a random θ, with options to sample a total random T and a total random π. Note that π is called `theta.startT` throughout the code.

```
params = genParams(m, k, backgroundAmount, L, order, true, true);
theta = genTheta(params, false, false);
```

- **HOP-HMM/src/mainRealData.m**

  Entry point of the code, reads data from the human genome, trains HOP-HMMs model and compares posterior probability to real epigenetic data. Execution of mainRealData will produce figures similar to figure 23

  ```
  mainRealData();
  ```

- **HOP-HMM/src/mainPosterior.m**

  Entry point of the code, follows the workflow of figure 17. Execution of mainPosterior plots random set of sequences with their Viterbi path and posterior probabilities similar to figure 21, and a confusion matrix similar to figure 22.

  ```
  mainPosterior();
  ```

- **HOP-HMM/src/mainDecErrorPlot.m**

  Entry point of the code, follows the workflow of figure 17. At each iteration of the EM, likelihood and errors are collected to form plots similar to figures 19 and 20.

  ```
  mainDecErrorPlot();
  ```

- **HOP-HMM/src/+EM/EM.m**

  The function actually trains the HOP-HMM from a given DNA sequence is the EM(). The neighboring code files residing in the +EM folder which contains it are the implementations of the E and M steps described in the introduction part of this work.

  ```
  [test, train] = misc.crossValidationSplit(params,
  mergedPeaksMin, testTrainRatio);
  [bestTheta, bestLikelihood, bestThetas] = EM(train, params,     maxIter,
  patience, repeat);
  ```

  where `maxIter` is the maximal number of iterations allowed in a run, `parience` is the number of iterations without likelihood increase allowed in a run and `repeat` is the number of different runs with different initializations which are tried.

6 Bibliography

Ahituv, N., Zhu, Y., Visel, A., Holt, A., Afzal, V., Pennacchio, L. A., & Rubin, E. M. (2007). Deletion of ultraconserved elements yields viable mice. PLoS biology, 5(9), e234.

Ainscough, R., Bardill, S., Barlow, K., Basham, V., Baynes, C., Beard, L., ... & Burrows, C. (1998). Genome sequence of the nematode C. elegans: a platform for investigating biology. Science, 282(5396), 2012-2018.

Alipanahi, B., Delong, A., Weirauch, M. T., & Frey, B. J. (2015). Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nature biotechnology, 33(8), 831.

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. The annals of mathematical statistics, 37(6), 1554-1563.

Bejerano, G., Pheasant, M., Makunin, I., Stephen, S., Kent, W. J., Mattick, J. S., & Haussler, D. (2004). Ultraconserved elements in the human genome. Science, 304(5675), 1321-1325.

Benko, S., Fantes, J. A., Amiel, J., Kleinjan, D., Thomas, S., Ramsay, J., et al. (2009). Highly conserved non. Nature Genetics 64(2), p. 10-12.

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

Burge, C., & Karlin, S. (1997). Prediction of complete gene structures in human genomic DNA. Journal of molecular biology, 268(1), 78-94.

Calo, E., & Wysocka, J. (2013). Modification of enhancer chromatin: what, how, and why?. Molecular cell, 49(5), 825-837.

Claverie, J. M. (1997). Computational methods for the identification of genes in vertebrate genomic sequences. Human molecular genetics, 6(10), 1735-1744.

Creyghton, M. P., Cheng, A. W., Welstead, G. G., Kooistra, T., Carey, B. W., Steine, E. J., ... & Boyer, L. A. (2010). Histone H3K27ac separates active from poised enhancers and predicts developmental state. Proceedings of the National Academy of Sciences, 107(50), 21931-21936.

Cutter, A. R., & Hayes, J. J. (2015). A brief review of nucleosome structure. FEBS letters, 589(20), 2914-2922.

De Beer, Z. W., Duong, T. A., Barnes, I., Wingfield, B. D., & Wingfield, M. J. (2014). Redefining Ceratocystis and allied genera. Studies in Mycology, 79, 187-219.

Doniger, S. W., Huh, J., & Fay, J. C. (2005). Identification of functional transcription factor binding sites using closely related Saccharomyces species. Genome research, 15(5), 701-709.

Dupin, M., Reynaud, P., Jarošík, V., Baker, R., Brunel, S., Eyre, D., ... & Makowski, D. (2011). Effects of the training dataset characteristics on the performance of nine species distribution models: application to Diabrotica virgifera virgifera. PLoS One, 6(6).

Du Preez, J. A. (1998). Efficient training of higher-order hidden Markov models using first-order representations. Computer speech & language, 12(1), 23-39.

Emison, E. S., McCallion, A. S., Kashuk, C. S., Bush, R. T., Grice, E., Lin, S., ... & Chakravarti, A. (2005). A common sex-dependent mutation in a RET enhancer underlies Hirschsprung disease risk. Nature, 434(7035), 857.

Ernst, J., & Kellis, M. (2012). ChromHMM: automating chromatin-state discovery and characterization. Nature methods, 9(3), 215.

Ernst, J., Kheradpour, P., Mikkelsen, T. S., Shoresh, N., Ward, L. D., Epstein, C. B., ... & Ku, M. (2011). Mapping and analysis of chromatin state dynamics in nine human cell types. Nature, 473(7345), 43.

Ezkurdia, I., Juan, D., Rodriguez, J. M., Frankish, A., Diekhans, M., Harrow, J., ... & Tress, M. L. (2014). Multiple evidence strands suggest that there may be as few as 19 000 human protein-coding genes. Human molecular genetics, 23(22), 5866-5878.

Ferguson, J. D. (1980). pp. 143–179, Variable duration models for speech. In Proc. of the Symposium on the applications of hidden Markov models to text and speech, JD Ferguson, Ed. Princeton: IDA-CRD.

Fishilevich, S., Nudel, R., Rappaport, N., Hadar, R., Plaschkes, I., Iny Stein, T., ... & Lancet, D. (2017). GeneHancer: genome-wide integration of enhancers and target genes in GeneCards. Database, 2017.

Friedli, M., Barde, I., Arcangeli, M., Verp, S., Quazzola, A., Zakany, J., ... & Duboule, D. (2010). A systematic enhancer screen using lentivector transgenesis identifies conserved and non-conserved functional elements at the Olig1 and Olig2 locus. PLoS One, 5(12), e15741.

Galperin, M. Y., & Fernández-Suarez, X. M. (2011). The 2012 nucleic acids research database issue and the online molecular biology database collection. Nucleic acids research, 40(D1), D1-D8.

Haussler, D. K. D., & Eeckman, M. G. R. F. H. (1996). A generalized hidden Markov model for the recognition of human genes in DNA. In Proc. int. conf. on intelligent systems for molecular biology, st. louis (pp. 134-142).

Hayashi-Takanaka, Y., Yamagata, K., Wakayama, T., Stasevich, T. J., Kainuma, T., Tsurimoto, T., ... & Kimura, H. (2011). Tracking epigenetic histone modifications in single cells using Fab-based live endogenous modification labeling. Nucleic acids research, 39(15), 6475-6488.

Heintzman, N. D., Stuart, R. K., Hon, G., Fu, Y., Ching, C. W., Hawkins, R. D., ... & Wang, W. (2007). Distinct and predictive chromatin signatures of transcriptional promoters and enhancers in the human genome. Nature genetics, 39(3), 311.

Heintzman, N. D., Hon, G. C., Hawkins, R. D., Kheradpour, P., Stark, A., Harp, L. F., ... & Ching, K. A. (2009). Histone modifications at human enhancers reflect global cell-type-specific gene expression. Nature, 459(7243), 108.

Hu, J., Brown, M. K., & Turin, W. (1996). HMM based online handwriting recognition. IEEE Transactions on pattern analysis and machine intelligence, 18(10), 1039-1045.

Jin Q, Yu L-R, Wang L, Zhang Z, Kasper LH, Lee J-E, Wang C, Brindle PK, Dent SYR, Ge K. 2011. Distinct roles of GCN5/PCAF-mediated H3K9ac and CBP/p300-mediated H3K18/27ac in nuclear receptor transactivation. The EMBO Journal 30:249–262.

Jones, P. A. (2012). Functions of DNA methylation: islands, start sites, gene bodies and beyond. Nature Reviews Genetics, 13(7), 484.

Kaplan, T., & Biggin, M. D. (2012). Quantitative models of the mechanisms that control genome-wide patterns of animal transcription factor binding. In Methods in cell biology (Vol. 110, pp. 263-283). Academic Press.

Karmodiya, K., Krebs, A. R., Oulad-Abdelghani, M., Kimura, H., & Tora, L. (2012). H3K9 and H3K14 acetylation co-occur at many gene regulatory elements, while H3K14ac marks a subset of inactive inducible promoters in mouse embryonic stem cells. BMC genomics, 13(1), 424.

Kelley, D. R., Snoek, J., & Rinn, J. L. (2016). Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. Genome research, 26(7), 990-999.

Khan, A., Fornes, O., Stigliani, A., Gheorghe, M., Castro-Mondragon, J. A., van der Lee, R., ... & Baranasic, D. (2017). JASPAR 2018: update of the open-access database of transcription factor binding profiles and its web framework. Nucleic acids research, 46(D1), D260-D266.

Kleftogiannis, D., Kalnis, P., Arner, E., & Bajic, V. B. (2016). Discriminative identification of transcriptional responses of promoters and enhancers after stimulus. Nucleic acids research, 45(4), e25-e25.

Kreimer, A., Zeng, H., Edwards, M. D., Guo, Y., Tian, K., Shin, S., ... & Li, Y. (2017). Predicting gene expression in massively parallel reporter assays: a comparative study. Human mutation, 38(9), 1240-1250.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2), 83-97.

Kulakovskiy, I. V., Belostotsky, A. A., Kasianov, A. S., Esipova, N. G., Medvedeva, Y. A., Eliseeva, I. A., & Makeev, V. J. (2011). A deeper look into transcription regulatory code by preferred pair distance templates for transcription factor binding sites. Bioinformatics, 27(19), 2621-2624.

Kundaje, A., Meuleman, W., Ernst, J., Bilenky, M., Yen, A., Heravi-Moussavi, A., ... & Amin, V. (2015). Integrative analysis of 111 reference human epigenomes. Nature, 518(7539), 317.

Lee, L. M., & Lee, J. C. (2006, June). A study on higher-order hidden Markov models and applications to speech recognition. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 682-690). Springer, Berlin, Heidelberg.

Lettice, L. A., Heaney, S. J., Purdie, L. A., Li, L., de Beer, P., Oostra, B. A., ... & de Graaff, E. (2003). A long-range Shh enhancer regulates expression in the developing limb and fin and is associated with preaxial polydactyly. Human molecular genetics, 12(14), 1725-1735.

Lindblad-Toh, K., Garber, M., Zuk, O., Lin, M. F., Parker, B. J., Washietl, S., ... & Ward, L. D. (2011). A high-resolution map of human evolutionary constraint using 29 mammals. Nature, 478(7370), 476.

Mari, J. F., Haton, J. P., & Kriouile, A. (1997). Automatic word recognition based on second-order hidden Markov models. IEEE Transactions on speech and Audio Processing, 5(1), 22-25.

Markov, A. A. (1906). Extension of the law of large numbers to dependent quantities. Izv. Fiz.-Matem. Obsch. Kazan Univ.(2nd Ser), 15, 135-156.

Miguel-Escalada, I., Pasquali, L., & Ferrer, J. (2015). Transcriptional enhancers: functional insights and role in human disease. Current opinion in genetics & development, 33, 71-76.

Ng, S. B., Turner, E. H., Robertson, P. D., Flygare, S. D., Bigham, A. W., Lee, C., ... & Bamshad, M. (2009). Targeted capture and massively parallel sequencing of 12 human exomes. Nature, 461(7261), 272.

Pennacchio, L. A., Ahituv, N., Moses, A. M., Prabhakar, S., Nobrega, M. A., Shoukry, M., ... & Plajzer-Frick, I. (2006). In vivo enhancer analysis of human conserved non-coding sequences. Nature, 444(7118), 499-502.

Pennacchio, L. A., Bickmore, W., Dean, A., Nobrega, M. A., & Bejerano, G. (2013). Enhancers: five essential questions. Nature Reviews Genetics, 14(4), 288.

Przybilla, J., Galle, J., & Rohlf, T. (2012). Is adult stem cell aging driven by conflicting modes of chromatin remodeling?. Bioessays, 34(10), 841-848.

Quinlan, A. R., & Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics, 26(6), 841-842.

Rabiner, L., & Juang, B. H. (1993). Fundamentals of speech processing. Prantice Hall.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257-286.

Rada-Iglesias, A., Bajpai, R., Swigut, T., Brugmann, S. A., Flynn, R. A., & Wysocka, J. (2011). A unique chromatin signature uncovers early developmental enhancers in humans. Nature, 470(7333), 279.

Rosin, J. M., Abassah-Oppong, S., & Cobb, J. (2013). Comparative transgenic analysis of enhancers from the human SHOX and mouse Shox2 genomic regions. Human molecular genetics, 22(15), 3063-3076.

Smemo, S., Campos, L. C., Moskowitz, I. P., Krieger, J. E., Pereira, A. C., & Nobrega, M. A. (2012). Regulatory variation in a TBX5 enhancer leads to isolated congenital heart disease. Human molecular genetics, 21(14), 3255-3263.

Soldner, F., Stelzer, Y., Shivalila, C. S., Abraham, B. J., Latourelle, J. C., Barrasa, M. I., ... & Jaenisch, R. (2016). Parkinson-associated risk variant in distal enhancer of α-synuclein modulates target gene expression. Nature, 533(7601), 95.

Stadler, M. B., Murr, R., Burger, L., Ivanek, R., Lienert, F., Schöler, A., ... & Tiwari, V. K. (2011). DNA-binding factors shape the mouse methylome at distal regulatory regions. Nature, 480(7378), 490.

Stormo, G. D., Schneider, T. D., Gold, L., & Ehrenfeucht, A. (1982). Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. Nucleic acids research, 10(9), 2997-3011.

Taher, L., McGaughey, D. M., Maragh, S., Aneas, I., Bessling, S. L., Miller, W., ... & Ovcharenko, I. (2011). Genome-wide identification of conserved regulatory function in diverged sequences. Genome research, 21(7), 1139-1149.

Tate, P. H., & Bird, A. P. (1993). Effects of DNA methylation on DNA-binding proteins and gene expression. Current opinion in genetics & development, 3(2), 226-231.

Thurman, R. E., Rynes, E., Humbert, R., Vierstra, J., Maurano, M. T., Haugen, E., ... & Garg, K. (2012). The accessible chromatin landscape of the human genome. Nature, 489(7414), 75.

Turin, W., & Sondhi, M. M. (1993). Modeling error sources in digital channels. IEEE Journal on Selected Areas in Communications, 11(3), 340-347.

Visel, A., Minovitsky, S., Dubchak, I., & Pennacchio, L. A. (2007). VISTA Enhancer Browser—a database of tissue-specific human enhancers. Nucleic Acids Research, 35(Database issue), D88.

Visel, A., Prabhakar, S., Akiyama, J. A., Shoukry, M., Lewis, K. D., Holt, A., ... & Pennacchio, L. A. (2008). Ultraconservation identifies a small subset of extremely constrained developmental enhancers. Nature genetics, 40(2), 158-160.

Visel, A., Blow, M. J., Li, Z., Zhang, T., Akiyama, J. A., Holt, A., ... & Afzal, V. (2009). ChIP-seq accurately predicts tissue-specific activity of enhancers. Nature, 457(7231), 854.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE transactions on Information Theory, 13(2), 260-269.

Williamson, I., Hill, R. E., & Bickmore, W. A. (2011). Enhancers: from developmental genetics to the genetics of common human disease. Developmental cell, 21(1), 17-19.

Winter, R. B., Berg, O. G., & Von Hippel, P. H. (1981). Diffusion-driven mechanisms of protein translocation on nucleic acids. 3. The Escherichia coli lac repressor-operator interaction: kinetic measurements and conclusions. Biochemistry, 20(24), 6961-6977.

Yang, F., Balakrishnan, S., & Wainwright, M. J. (2015, December). Statistical and computational guarantees for the Baum-Welch algorithm. In 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton) (pp. 658-665). IEEE.

Zentner, G. E., Tesar, P. J., & Scacheri, P. C. (2011). Epigenetic signatures distinguish multiple classes of enhancers with distinct cellular functions. Genome research, 21(8), 1273-1283.

Zhang, Y., Liu, T., Meyer, C. A., Eeckhoute, J., Johnson, D. S., Bernstein, B. E., ... & Liu, X. S. (2008). Model-based analysis of ChIP-Seq (MACS). Genome biology, 9(9), R137.

Zhou, J., & Troyanskaya, O. G. (2015). Predicting effects of noncoding variants with deep learning–based sequence model. Nature methods, 12(10), 931.