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

by

David Taub

Supervised by

Prof. Tommy Kaplan

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

April 2020

The Faculty of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel

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 rules and nuances of enhancer structure is not fully understood yet, though it has been shown that transcription factors tend to attach to them at unique motifs called transcription factor binding sites, which are over-represented in enhancer sequences. 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 spatial 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 out of 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 sequence in order to regulate its gene expression. With that idea in mind, we offer a computational approach for the detection of enhancers based on their sequences alone, and in an unsupervised manner. We created a high-order generalized 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'll first review the biological background of enhancers, specifically in humans. Then we'll review in depth the background of Markov and hidden Markov models, and discuss how to calculate the likelihood of a sequence given this model. We'll 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. These algorithms implementations are demonstrated by applying them to a synthetic dataset of enhancer-like sequences created by using the generative property of the generalized model. We simulate the model in a controlled way to evaluate its performance by inferring estimated parameters of the model and comparing them to the real parameters used to create the dataset. Finally, we apply the expectation maximization algorithm for training a HOP-HMM from human DNA enhancer sequences, selected by the epigenetic data of the Roadmap project. We demonstrate the capabilities of the model by comparing its estimation to the epigenetic tracks, showing it can predict the loci of enhancers and in which tissues they will be active, without exposure to epigenetic data.

Introduction

The genome of every living organism contains the inherited information which defines its complex structure and function. The genome is built out of deoxyribonucleic acid (DNA) molecules, a structure of two chains of nucleotides units forming a double helix shape. Nucleotides are built out of 4 different basic elements: cytosine, guanine, adenine or thymine or in short A,C,G and T. 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 ensure various roles and functions within organisms. They have the structure of a polymer built out of 20 different amino acids, whose order and structure 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. In the transcription process the gene is read and transcribed into a single strand sequence of RNA. Later, the formed RNA molecule, 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 a sequence of amino acids constituting the protein.

Gene sequences are built out of fragmented introns and exons, where only the exons mature into mRNA molecules which are translated into proteins, while the introns are spliced away beforehand. Counter intuitively, even though the exons hold the recipe for the construction of the proteins of the organism, its complexity is not a product of their number or their length. 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), even though the human body is much more diverse and complex. Although the genes are responsible for the variety of proteins a cell can produce, the source of organism complexity, with different cells performing different tasks while carrying the same genome, stems from the gene regulation mechanism. In the case of humans, the genome is 3.23 Gb long and it is estimated that the total length of gene regulation regions involves 10-20% of it (Pennacchio et al., 2015), compared to exon regions which involve only 1% (Ng et al., 2009).

Enhancers are non-coding regulatory DNA sequences which play a key role in the regulation transcription of genes. 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 activated, the DNA folding draws the enhancer 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 is the enhancer's target gene, and it can be located up to one megabase pair (Mb) upstream or downstream from its activating enhancer as 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).

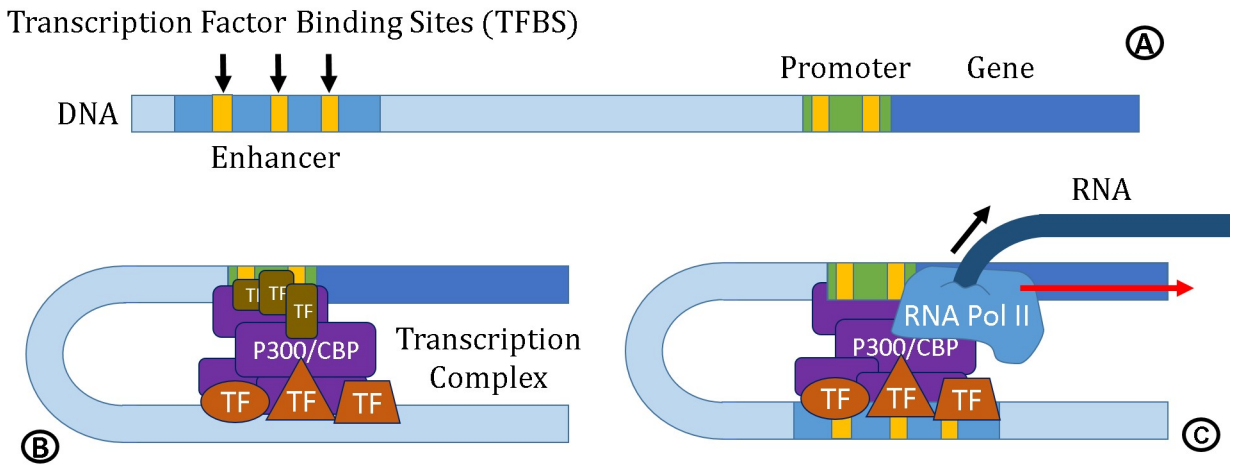


Figure 1: **A)** An enhancer and its distal target gene. **B)** The DNA folds and the attaches to transcription factors, which then draw other co-factor proteins that together form the transcription complex. **C)** The RNA Polymerase II is recruited and while moving along the gene, it generates a new RNA molecule which is transcribed off the gene.

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 the VISTA Project (Visel et al., 2007), fertilized mouse eggs were injected with enhancer sequences adjacent to a LacZ reporter gene, encoding an enzyme protein with a blue color. Since they were synthesized, the injected DNA sequences containing the enhancer and reporter genes bore no epigenetic information, and they were integrated into the mouse genome in an arbitral position. The enhancers in the injected DNA sequences originated from the human genome, and each enhancer was injected into a different embryo. 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.

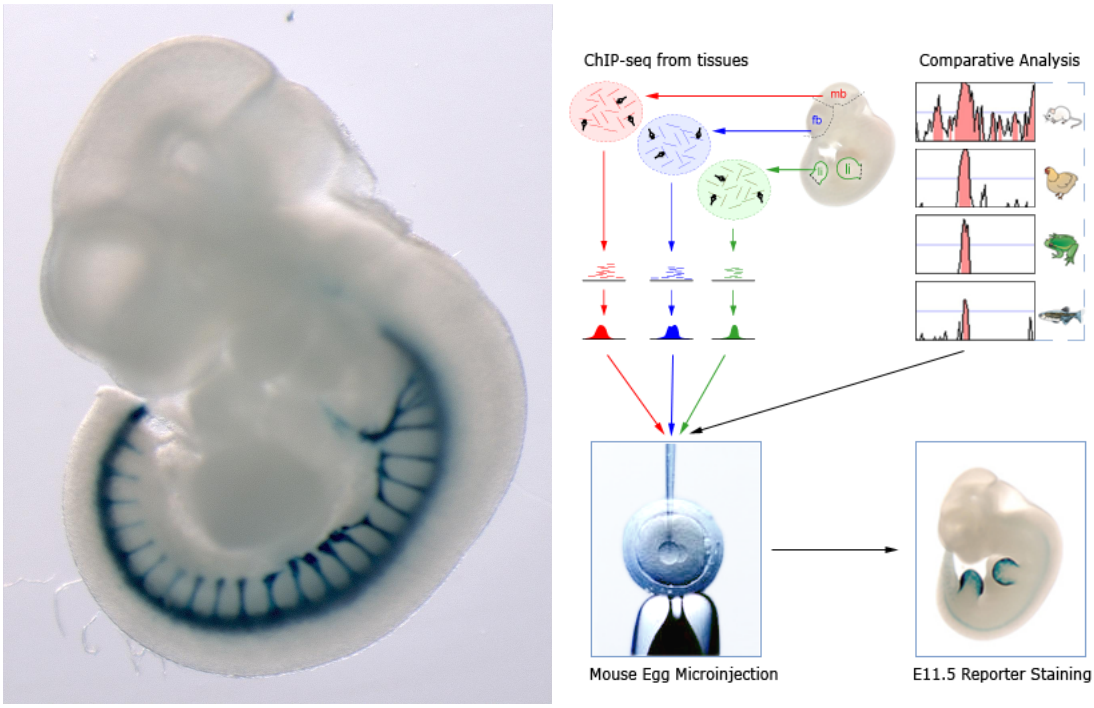


Figure 2: Transgenic mouse embryo on the 11.5 day. A fertilized egg was injected with a synthetic enhancer sequence known to be related to the dorsal root ganglia of spinal neurons. The enhancer became activated and caused the expression of the blue color marker gene that was coupled to it. Both images are taken from Vista Enhancer Browser, on the left is experiment hs-51 embryo 2.

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 TFBS 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 interaction density, a chromatin immunoprecipitation sequencing (ChIP-seq) method is used to probe the amount of TFs in affinity to the DNA strands. Briefly, this method involved applying antibodies on cross-linked DNA, which attach to the TFs linked to the DNA. This antibody attachment is followed by massive parallel sequencing of the short DNA sequences around the TF and the antibody. 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 enhancer's TFBSs 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 S, 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), and 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 in Stormo et al. (1982), is the most commonly used probabilistic model to address this task. The underlying assumption of the PWM model is that every position in the DNA sequence has an independent probability to attach to the TF, and therefore the total binding probability is a multiplication of all the per-position probabilities in the motif:

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

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 TF's ChIP-seq peaks. For each j, $P(x_j|binding)$) 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 is stored in a position probability matrix (PPM) 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 lengths, the use of the normalized form of PPM, the PWM, is more convenient:

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

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 convolution of M is applied on 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. In this work we'll use the more familiar term PWMs though we actually used the unnormalized PPMs for the TFBS emission model, since we required the likelihood of a TF binding and not a length-independent comparison between TFs.

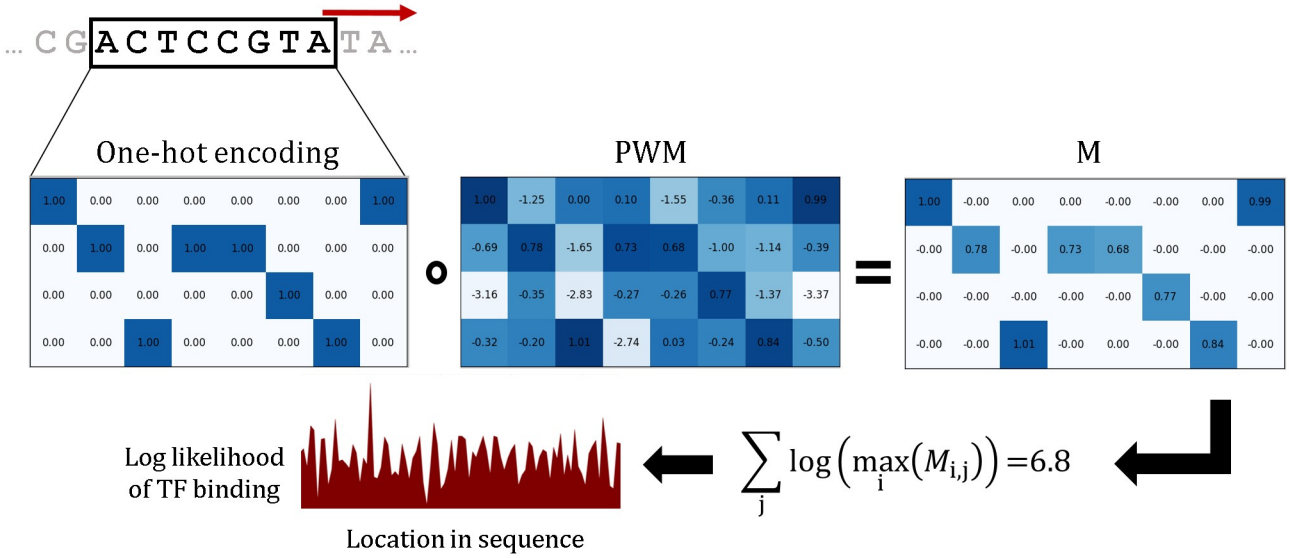


Figure 3: Sub-sequences out of the DNA are represented in a one-hot encoding, and multiplied entry-wise by a PWM. Then, the sum of the logs of the maximal values in each column of the resulting matrix is calculated, which represents the log likelihood of the TF binding to the sub-sequence. This log likelihood is calculated for each location in the sequence, in which locations with high values indicate a high likelihood of TF binding.

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 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 in (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 to regulatory activity, but their deletion still yields viable mice (Ahituv et al., 2007). Nevertheless, an assay of elements with 100% sequence identity of over 200 bp between human and mouse found that 50% showed enhancer activity in mice (Visel et al., 2007). 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, which do not involve the nucleotide sequences themselves, 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. A single cell has its own epigenetic features, often in binary form, e.g. a specific element of the genome can be either accessible or not. When several cells epigenetic properties are measured, usually a frequency or count of the measured feature per DNA locus is calculated along the reference genome. The epigenetic data is commonly further processed by calculating its 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, and they form 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 et al., 2015). The DNA wrapped around a nucleosome has a lesser likelihood for 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 (Buenrostro et al., 2013), and is currently more commonly used.

Histone modifications, also called histone marks and 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 will be 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; Fernandez and Galperin, 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 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 attach 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 (Stormo et al., 1982), and is thought to precede the H3K27ac modification (Staden, 1984; 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 mark for the differentiation of active enhancers from poised enhancers (Staden, 1984), 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 H3K4me1and H3K27me3 are typical of poised enhancers (Staden, 1984).

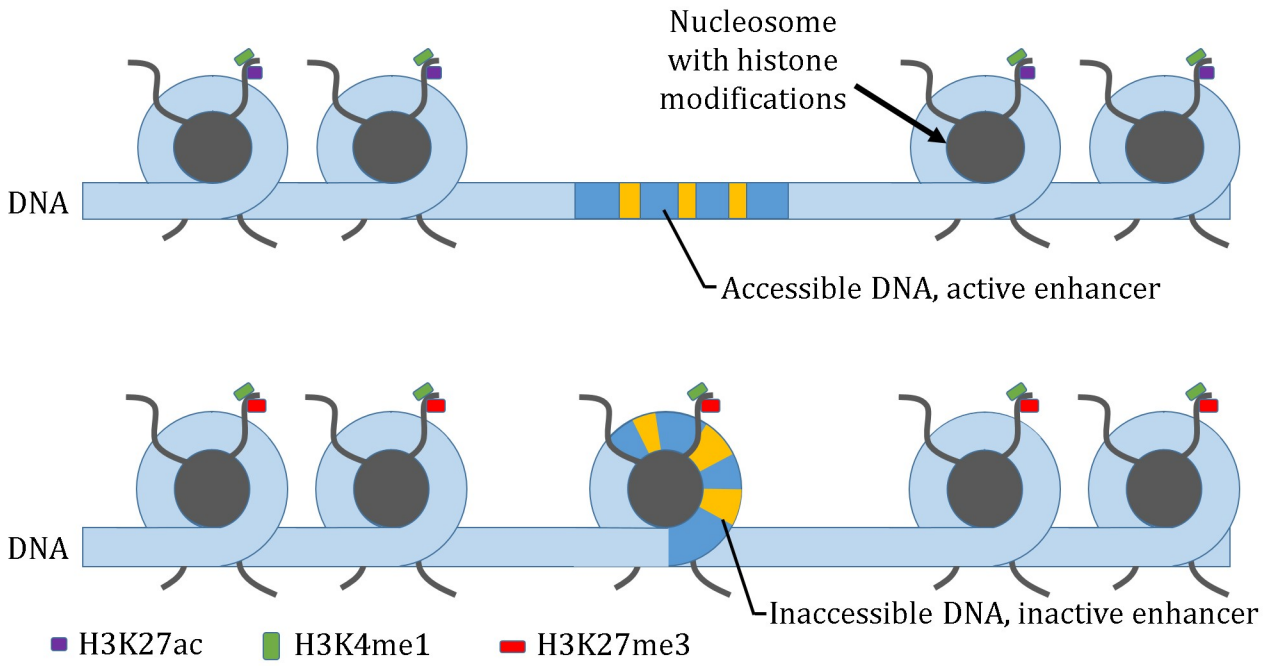


Figure 4: The accessibility of an enhancer's sequence and its surrounding histone modifications are connected to its regulatory activity state. The upper diagram shows an active enhancer sequence accessible to the protein interaction needed for transcription, whereas the lower one shows an inactive enhancer 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 et al., 2012) and 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 cause and consequence relationships underlying these correlations is not yet clear. Currently, the most accurate method for the 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).

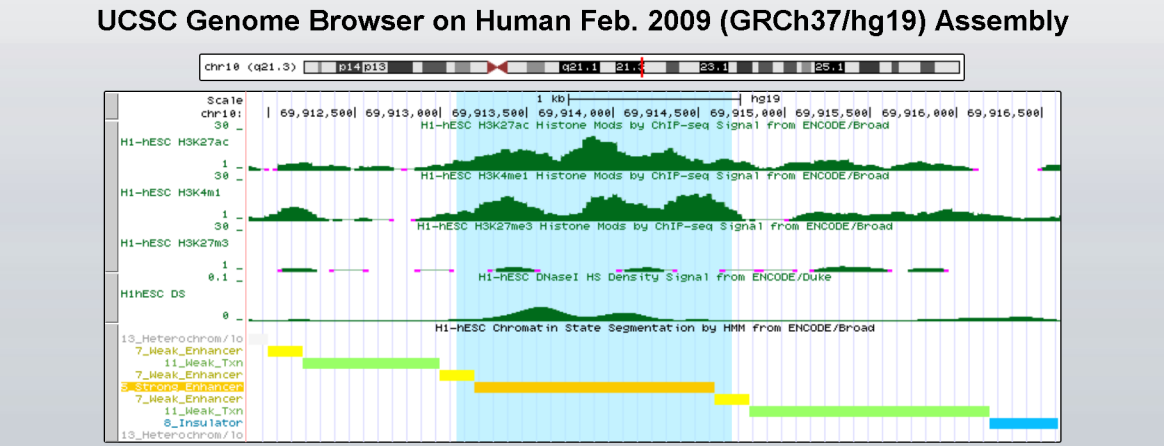


Figure 5: UCSC Genome Browser showing epigenetic features tracks, taken from the 10th chromosome of a H1-hESC cell line. Highlighted in light blue, the peaks of H3K27ac (1st green plot) and H3K4me1 (2nd green plot) histone modifications and the DNase-I hypersensitivity features (4th green plot), together 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 in between two nucleosomes with modifications. Taken from `https://genome-euro.ucsc.edu/cgi-bin/hgTracks`

Related Work

Several significant computational efforts were made in the last few years for predicting the epigenetic and regulatory properties of DNA elements based on the genetic sequence alone. DeepSEA (Zhou and Troyanskaya, 2015) uses 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 are the chromatin modifications extracted from ENCODE and Roadmap epigenetic data releases. Basset (Khan et al., 2018) 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 binding of 538 TFs and 194 RNA binding proteins from DNA sequences of varying lengths. In gkm-SVM (Beer et al., 2014), gapped *k*-mers presence indicator vectors were used as features for a SVM classifier in order to predict the role of DNA sequences of varying lengths. ChromHMM (Ernst and Kellis, 2012) is a widely used software that tackles the problem of analyzing the epigenetic data to predict the role of fragments of genomic sequence. The algorithm converts to binary the chromatin modification values by whether or not it exceeded a threshold, which is then inserted as input to HMM that classifies the genome states. A 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 letters 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 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. In this technique, the order of the k-mer is sacrificed for a more meaning-oriented, structured and fixed-length data encoding, similarly to the bag of words technique in text analysis and natural language processing.

Machine Learning Models

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 deciding its role label Y for a given 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 from it $P(Y|X)$. 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 conditional probability $P(Y|X)$, which is estimated directly from the dataset.

Both models eventually base their classification 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, 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 was also used for the task of classifying the role of DNA sequences. As a discriminative model it assumes very little regarding the way the DNA sequence is generated based on its role, but finds instead features in the sequence that imply its role. Hence it is often difficult to use such a model for a 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 of transitions between the states being denoted by $T_{i,j} = P(y_t = S_i|y_{t-1} = S_j)$. 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:

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

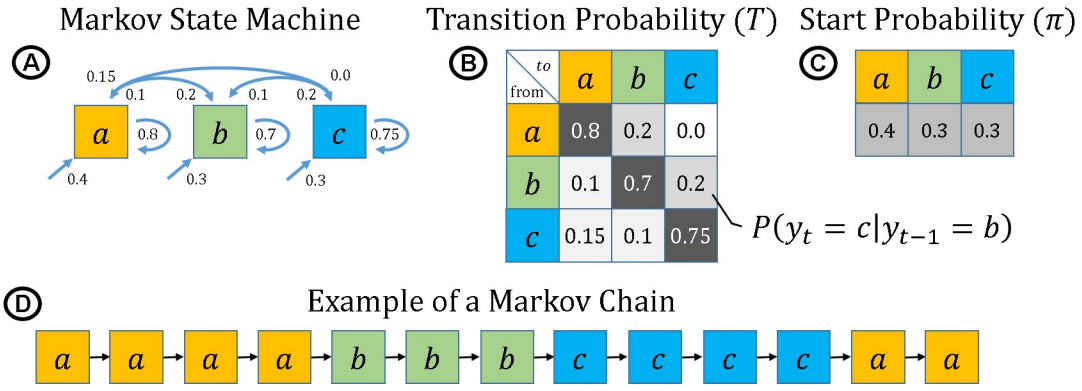


Figure 6: **A)** Markov model with 3 states (a,b and c). **B,C)** The model starts with a state sampled from π, and travels between the states with a transition distribution T. **D)** The model can generate Markov chains of states, where the transition between the states is only conditioned by the previous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

The hidden Markov model (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 called 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 $E_{i,j} = P(x_t = V_j | y_t = S_i)$. Similarly to the Markov model, the distribution to the first hidden state is marked as π and the transition distribution is marked as T.

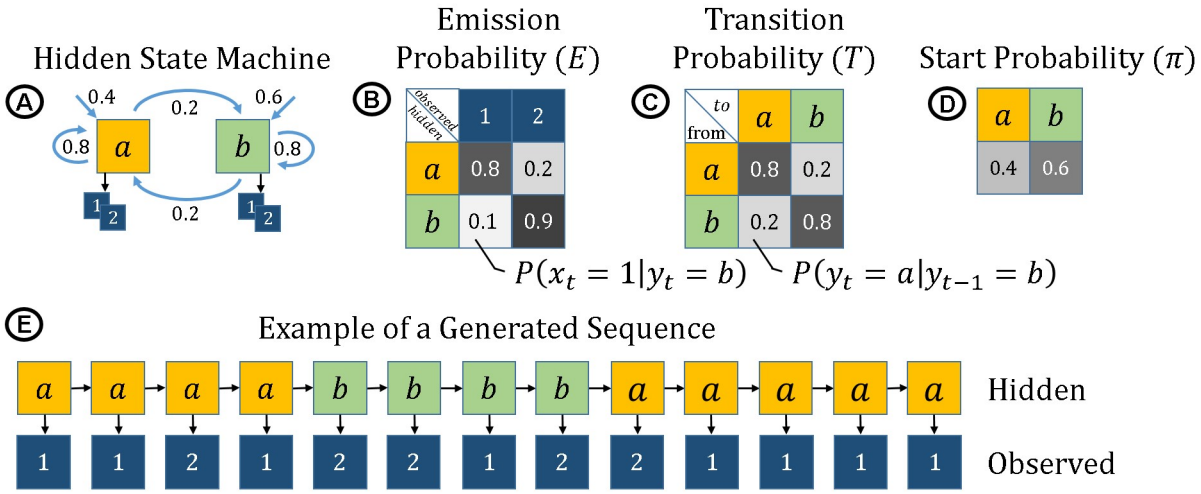


Figure 7: **A)** HMM with 2 hidden states. **B)** The observed variables (dark blue) are emitted by the hidden state at their location, sampled from the discrete conditional distribution E. **C,D)** The hidden states (yellow and green) behave as Markov model states with starting and transition probabilities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

HMM is a very popular signal processing algorithm that has been adopted in the various fields of computational biology since the 1980's. HMM was proposed by Leonard Baum (Baum et al., 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 where the observable sequence is made out of DNA, a simplistic model can assume that the DNA sequence is composed out of 4 states: genes, promoter enhancers and background regions. Each of these states will have a different nucleotide frequency, and we assume that the DNA sequence was generated by a 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 y1:L that describes the "mode" of the sequence for each location.

Having a 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 $P_\theta(x_{1:L})$?
- What is the probability of a hidden state at every location or $P_\theta(y_t = j | x_{1:L})$?

The two above-mentioned probabilities are named the likelihood function and the posterior probabilities of HMM. As in many generative models, HMM's 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:

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

The probability $P_\theta(x_{1:L})$ is called the incomplete-data likelihood function and the probability $P_\theta(x_{1:L}, y_{1:L})$ is called the complete-data likelihood function. In the case of HMM with parameters θ, the complete-data can be calculated by:

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

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 approach to overcome this gap uses the Markovian memorylessness of HMM, and answers both the likelihood and the posterior questions we raised above. This approach is called Forward-Backward algorithm: it was suggested as a step in the Baum-Welch algorithm (Baum et al. (1966)), which is an EM algorithm for finding the 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 dynamically calculated, holding the probabilities:

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

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

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:

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

It is calculated by the dynamic algorithm:

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.

$$\alpha_{j,t} = P_\theta(y_t = j, x_{1:t}) = P_\theta(x_t | y_t = j, x_{1:t-1}) \cdot P_\theta(y_t = j, x_{1:t-1}) =$$

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

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

Forward Algorithm

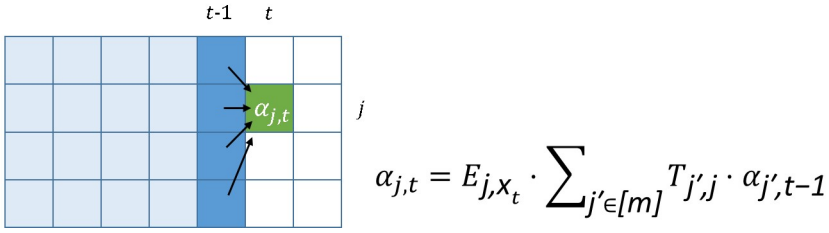


Figure 8: Forward algorithm dynamically calculates the probability stored in $\alpha_{j,t}$ by using the previously calculated $\alpha_{j',t-1}$ values.

Backward Algorithm

The backwards 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:

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

It is calculated by the dynamic algorithm:

Algorithm 2 Backward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

$\beta_{1:m,L} = 1$

$for\, t = [L-1,...,1]:$

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

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

This matrix building process is similarly explained by:

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

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

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

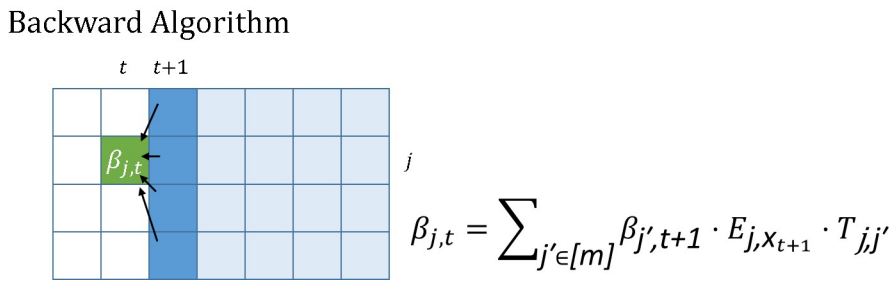


Figure 9: Backward algorithm dynamically calculates the probability stored in $\beta_{j,t}$ by using the previously calculated $\beta_{j',t+1}$ values

Once we obtain α and β probabilities, the incomplete-data likelihood of HMM can finally be easily calculated:

$$P_\theta(x_{1:L}) = \sum_{j\in[m]} P_\theta(y_L = j, x_{1:L}) = \sum_{j\in[m]} \alpha_{j,L} \tag{4}$$

And so can the posterior probability be computed:

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

Although HMM is simple and efficient, applying it on DNA sequences has a major disadvantage: the inherit Markovian lack-of-memory property. That is, the model's next state is always dependent only on the previous state, without further history consideration. For the task of emitting a TFBS motif, where each position has a different emission distribution depending on the location in the motif, a HMM would need to different hidden states per position 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 that require learning a large number of parameters, e.g. for the ability to emit 50 motifs of length 5, an HMM needs to have over 60,000 parameters. Furthermore, the enhancer modeling task at hand is even more complex, since we would like to model multiple enhancers and backgrounds states, each having different probability of emitting motifs and unique k-order emission distribution when not in those motifs. For our data structure prior assumption the required number of model's parameters would have been about 10^7, large enough to introduce problems such as unfeasible memory complexity and overfitting.

A common way to avoid overfitting the data when training machine learning models is reducing the model's complexity by fixing some of its parameters. Our proposed HOP-HMM addresses both the memory issue and the overfitting issue while remaining equivalent to a regularized HMM with a large number of states with fixed parameters. Namely, most of the transition probabilities are fixed to zero and therefore never stored in memory, and some of the emission probabilities are predetermined and are fixed during the training. This allows us to learn a model with the enhancer prior assumptions of motifs and high-order emission without overfitting, and with reasonable memory complexity.

Generalized HMM

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 genes prediction in the 1990's (Haussler and Eeckman (1996); Burge and Karlin (1997)), where specific exon states emitted codons instead of single nucleotides and feed forward neural networks were used for evaluating the probability of certain transitions.

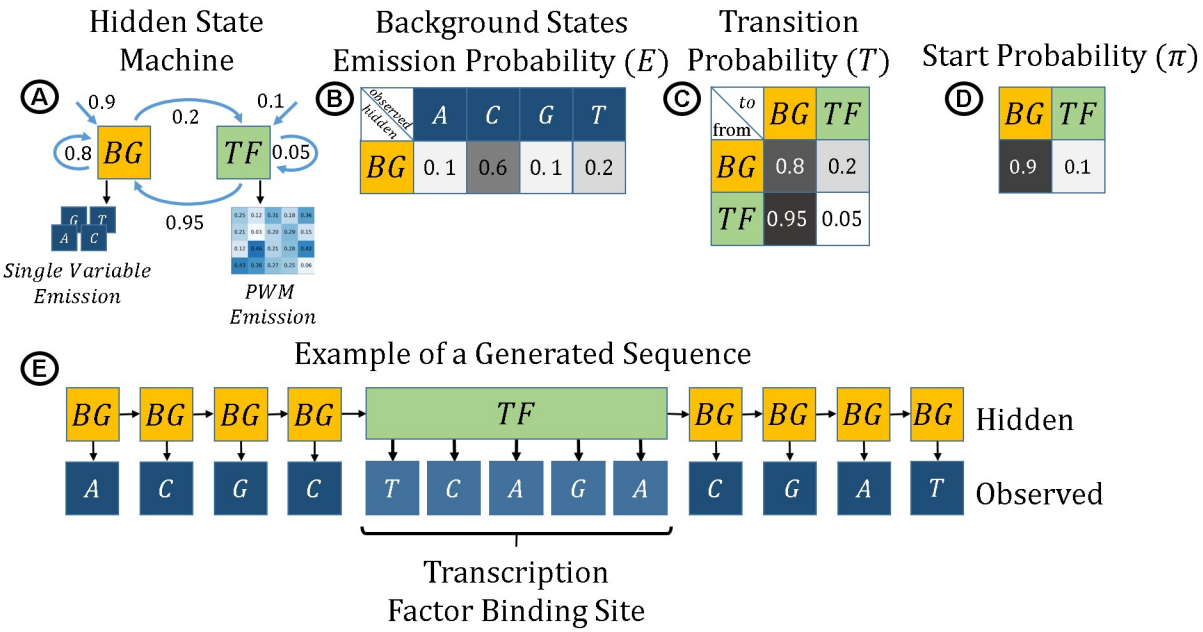


Figure 10: **A)** GHMM with a TF-state which emits using a PWM. The model has one background hidden state (yellow) and one TF hidden state. **B,C,D)** Although the TF-state emits motifs with 5 bp, the rest of the emissions, transitions and start probabilities remain the same as in a regular HMM. **E)** 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 called higher-order HMM uses conditional distribution by making the transition and emission dependent on previous hidden states (Ferguson, 1980; Mari et al., 1997; 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 higher-order emission which depends on the previous hidden states was previous used, high-order emission which depends on previous emitted observable variables is a less researched field. Such a HMM variant fits better to the locality nature for the task of emission k-mer structures, but it only requires $O\left(m^2+4^K\right)$ 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.

HOP-HMM

HOP-HMM is a GHMM that is well fitted to utilize the structure of enhancers containing TFBSs inside them, due to the TFBS emitting TF-states that take part in the generation process of the sequence. Due to the assumed local physical nature of the TF binding of the DNA sequence and success of HMM in gene prediction, we think the memorylessness of HMMs fit well to the task of enhancer prediction. HOP-HMM balances between the Markovian memorylessness and the observed k-mer of the TFBS present in regulatory regions in the DNA.

HOP-HMM extends the GHMM model of Kaplan et al., 2012, where some of the hidden states emit TFBS sampled from PWMs to predict enhancers location in the genome. In HOP-HMM we added the high order conditional emission probability on non-TF-states.

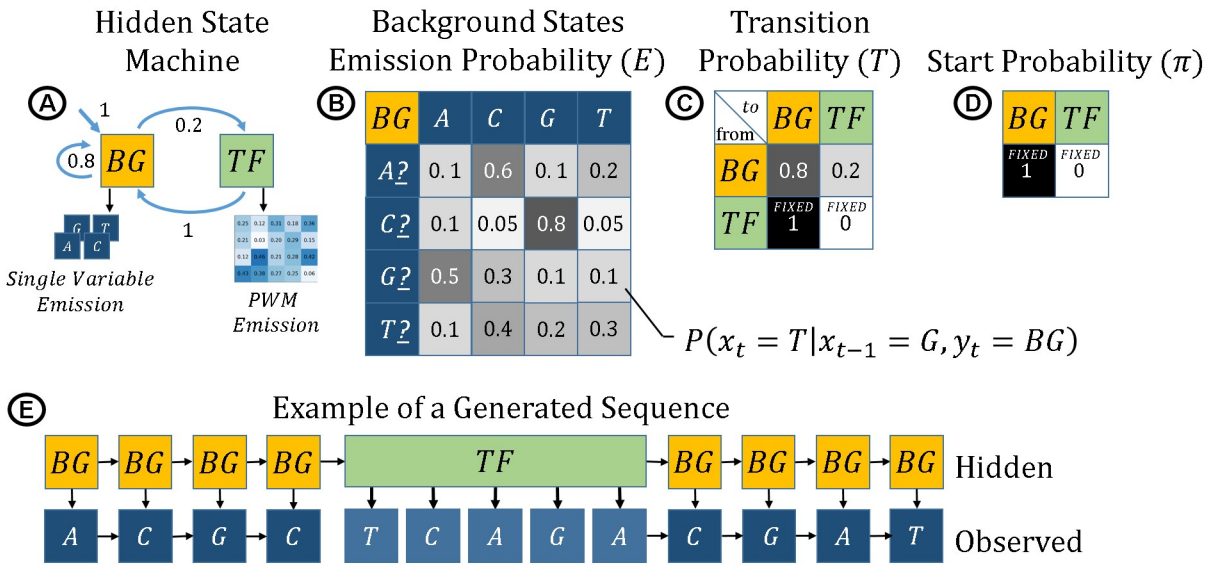


Figure 11: **A)** Small HOP-HMM that has one background-state and one TF-state. **B)** The background-state emits a single observable variable, and is has 2-order emission, meaning it is conditioned on the previous observable variable. **C,D)** Unlike GHMM, in HOP-HMM TF-state can transition into itself and cannot be the starting hidden state and the background-state. **E)** The TF-state emits multiple observable variable that represent a TFBS sampled from a PWM.

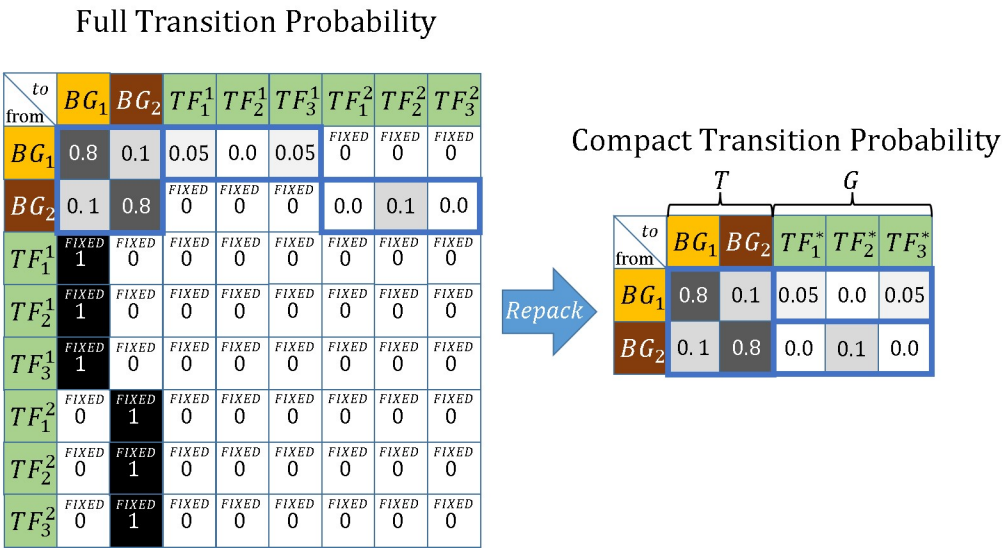


Figure 12: Instead of holding a single sparse 8×8 transition matrix, an alternative compact form holds only the non-fixed transition probabilities, split into T and G matrices. The non-fixed transition probabilities held in the compact form are the ones in between background-states, and between background-states to their TF-states (outlined with blue). A concatenation of a row in T and G holds the probability of the next hidden-state given the current background-state.

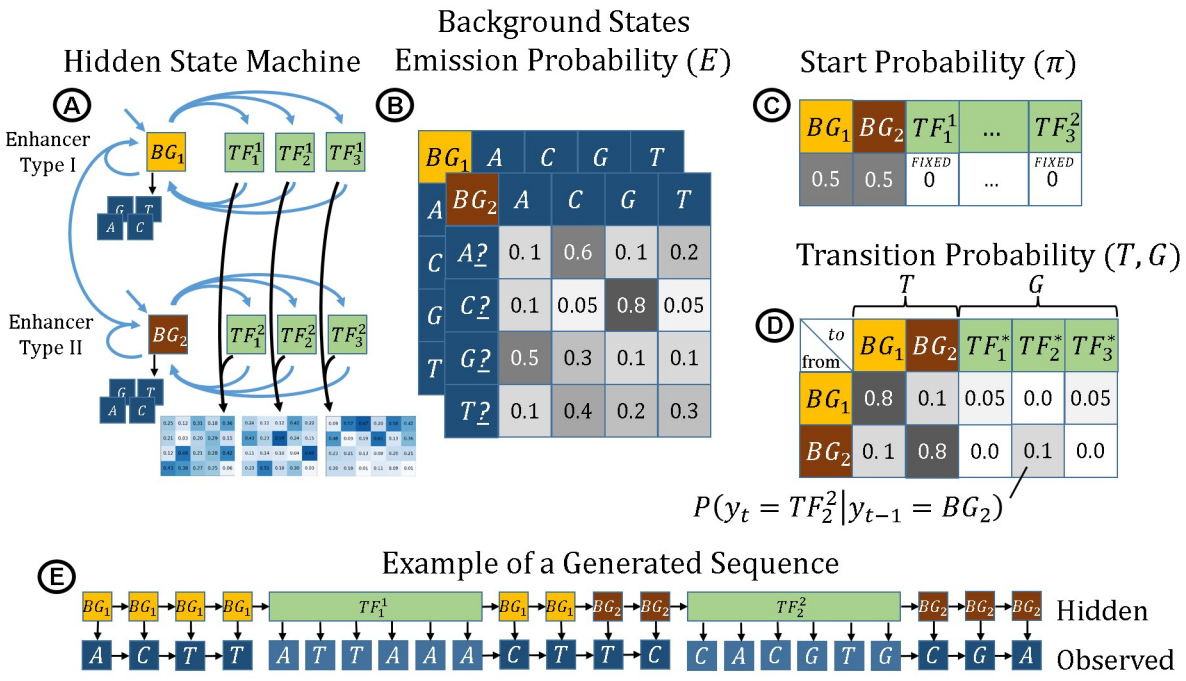


Figure 13: **A)** A more complex HOP-HMM with two background-state BG_1 and BG_2, where each has 3 TF-states. **B)** each of the background-states has its own 2-order emission distribution in a 4x4 matrix. **C)** The start hidden state distribution π allows only background-states to start the hidden sequence. **D)** The transition probability is held by matrices T and G. **E)** The example generated sequence is built out of two types of sequences, each has 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 background-states.
- TF-states are indexed as (j,l) where $j \in [m]$, $l \in [k]$. and k is the number of TF-states each of the background-states has.

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

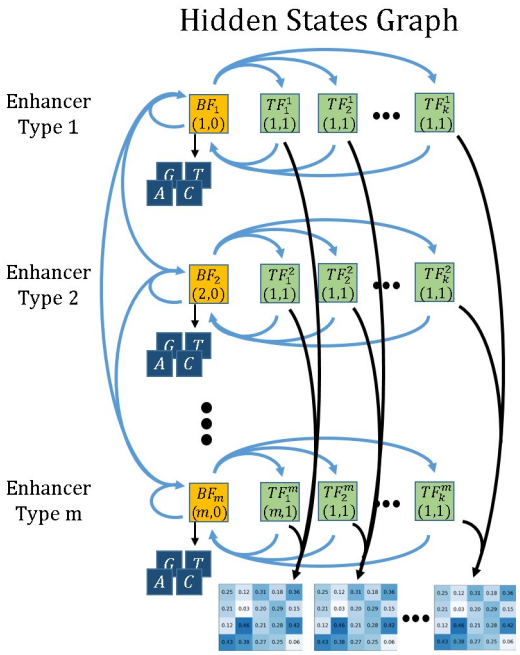


Figure 14: General hidden states graph of HOP-HMM. Each row represents a sequence type, where each of the m background-states (yellow) has k TF-states (green). Not all transitions are possible, moving between the rows is possible only by a background-state to background-state transition.

While most background-states $(j,0)$ represent an enhancer type, we also would like to model true background regions in between the enhancer that carry no regulatory role and have no TFBSs. For 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), where each column represents a nucleotide distribution at that position. When the model enters a TF-state, it emits a motif by sampling from a PWM column by column independently, 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, where their emission is conditional on the previous of letters emitted in the DNA sequence. The emission from background-states is done by sampling a nucleotide from the distributions stored in E tensor. E dimension is o+1, and its size is $m \times 4 \times 4 \times ... \times 4$ (with o fours) and its values are describe the emission probability $E_{j,x_{t-o+1},x_{t-o+2},...,x_t} = P(x_t | y_t = (j,0), x_{t-o+1}, ..., x_{t-1})$, meaning that when x_t is sampled by the model, the preceding $o-1$ observed variables are used as indices of the tensor for getting emission probability 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 vector, e.g. if $t = o-1$, a single variable is missing for emitting x_t and the distribution used for emission sampling 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. The first background-state, as in HMM, 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 transition are not allowed, a single transition matrix from all states to all states would be sparse. Instead, as described in figure 12, we hold only the possible transition probabilities in two matrices, representing the two types of allowed transitions:

- T for background-state to background-state transitions, a $m \times m$ matrix where $T_{j_1,j_2} = P(y_{t+1} = (j_2,0)|y_t = (j_1,0))$.
- G for background-state to TF-state transitions a $m \times k$ matrix where $G_{j,l} = P\left(y_{t+1:t+|W_l|} = (j,l)|y_t = (j,0)\right)$.

When in a background-state, after the observable variable emission, the model samples its next hidden state from a probability vector that is a concatenation of a row in T and a row in G. If a TF-state is chosen, after the TF-state's motif emission, the model returns back to the background-state to emit another single observable variable and so on.

Methods

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

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

Even though the incomplete-data likelihood of HMM in (2) is derivable by θ, optimizing it is as difficult as calculating it and therefore is also impractical. Instead, the strategy of the EM algorithm is to 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's parameters from previous EM iteration (or guessed parameters in the first iteration) and while assuming a fixed observed $x_{1:L}$, as it is the given DNA sequence. For this task we define our target function Q:

$$Q\left(\theta,\theta^{'}\right) = E_Y\left[log\left(P_\theta\left(x_{1:L},y_{1:L}\right)\right)|x_{1:L},\theta^{'}\right] = \sum_{y_{1:L}\in[m]^L} log\left(P_\theta\left(x_{1:L},y_{1:L}\right)\right)\cdot P_{\theta'}\left(x_{1:L},y_{1:L}\right) \tag{5}$$

Here E is expressing an expected value, not to be confused with the HMM emission probability. 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 the θ for maximizing $Q\left(\theta,\theta^{'}\right)$ in every M-step of the EM algorithm until convergence.

Using (3) we will split the Q function (5) into three independent parts:

$$\begin{aligned}
Q\left(\theta,\theta^{'}\right) = &\sum_{y_{1:L}\in[m]^L} log\pi_{y_1}\cdot P_{\theta'}\left(x_{1:L},y_{1:L}\right) \\
&+ \sum_{y_{1:L}\in[m]^L}\left(\sum_{t\in 2\ldots L} logT_{y_{t-1},y_t}\right)\cdot P_{\theta'}\left(x_{1:L},y_{1:L}\right) \\
&+ \sum_{y_{1:L}\in[m]^L}\left(\sum_{t\in[L]} logE_{y_t,x_t}\right)\cdot P_{\theta'}\left(x_{1:L},y_{1:L}\right)
\end{aligned}$$

then by manipulating the summation, the exponential state sequence summation could be simplified to:

$$\begin{aligned}
Q\left(\theta,\theta^{'}\right) = &\sum_{j\in[m]} log\pi_j\cdot P_{\theta'}\left(x_{1:L},y_1=j\right) \\
&+ \sum_{t\in 2\ldots L}\sum_{j_1,j_2\in[m]} logT_{j_1,j_2}\cdot P_{\theta'}\left(x_{1:L},y_{t-1}=j_1,y_t=j_2\right) \\
&+ \sum_{t\in[L]}\sum_{j\in[m]} logE_{j,x_t}\cdot P_{\theta'}\left(x_{1:L},y_t=j\right)
\end{aligned}$$

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

- $\sum_{j\in[m]} \pi_j = 1$
- $\sum_{j_2\in[m]} T_{j_1,j_2} = 1$ 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:

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

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

$$\pi_j = \frac{P_{\theta'}(x_{1:L}, y_1 = j)}{P_{\theta'}(x_{1:L})} = P_{\theta'}(y_1 = j | x_{1:L}) \tag{6}$$

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

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

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

when the m equations are summed, gives $\lambda_{j_1} = \sum_{t\in 2\ldots L} P_{\theta'}(x_{1:L}, y_{t-1} = j_1)$

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

$$T_{j_1,j_2} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}(x_{1:L}, y_{t-1} = j_1, y_t = j_2)}{\sum_{t\in 2\ldots L} P_{\theta'}(x_{1:L}, y_{t-1} = j_1)} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}(y_{t-1} = j_1, y_t = j_2 | x_{1:L})}{\sum_{t\in 2\ldots L} P_{\theta'}(y_{t-1} = j_1 | x_{1:L})} \tag{7}$$

Finally, we'll define multiplier λ_j for every $j \in [m]$ for the E part:

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

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

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

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

$$E_{j,b} = \frac{\sum_{t\in[L]} P_{\theta'}(x_{1:L}, y_t = j)\cdot 1_b(x_t)}{\sum_{t\in[L]} P_{\theta'}(x_{1:L}, y_t = j)} = \frac{\sum_{t\in[L]} P_{\theta'}(y_t = j|x_{1:L})\, 1_b(x_t)}{\sum_{t\in[L]} P_{\theta'}(y_t = j|x_{1:L})} \tag{8}$$

For us to be able to calculate these reestimation of θ as written in (6), (7) and (8), we are still left with the calculation of the two probabilities terms inside them. To resemble the notations coined in Rabiner (1989), the first widely accepted HMM application, we'll denote these as γ and ξ

$$\gamma_{t,j} = P_{\theta'}(y_t = j|x_{1:L}) \tag{9}$$

$$\xi_{t,j_1,j_2} = P_{\theta'}(y_{t-1} = j_1, y_t = j_2|x_{1:L}) \tag{10}$$

We will use (4) and the output of the Forward-Backward algorithm α and β for their calculation:

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

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

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

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

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

Baum-Welch Algorithm for HOP-HMM

The transitions and emissions mechanisms of HOP-HMM are different, and therefore the complete-data likelihood of HOP-HMM requires 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 (5):

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

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

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

The θ^* which optimizes Q here, $\theta^* = argmax_\theta Q(\theta,\theta')$, is archived by optimizing its 3 independent parts as well, each having its own constrain 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 $j \in [n]$

M-step

The π and E conditions produce almost exact same maximization as in regular Baum-Welch (6) and (8):

$$\pi_j = \frac{P_{\theta'}\left(x_{1:L}, y_1 = (j,0)|\theta'\right)}{P_{\theta'}\left(x_{1:L}|\theta'\right)} = P_{\theta'}\left(y_1 = (j,0)|x_{1:L}\right) \tag{11}$$

$$E_{j,b_1,\ldots,b_o} = \frac{\sum_{t\in o,\ldots,L} P_{\theta'}\left(x_{1:L}, y_t = (j,0)\right) 1_{b_1,\ldots,b_o}(x_{t-o+1,\ldots,t})}{\sum_{t\in o,\ldots,L} P_{\theta'}\left(x_{1:L}, y_t = (j,0)\right)} \tag{12}$$

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:

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

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

which yields $\lambda_{j_1} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}(x_{1:L}, y_{t-1}=(j_1,0), y_t=(j_2,0))}{T_{j_1,j_2}}$ and $\lambda_{j_1} = \frac{\sum_{t\in 2\ldots L}\cdot P_{\theta'}(x_{1:L}, y_{t-1}=(j_1,0), y_t=(j_1,l))}{G_{j_1 l}}$ for $j_2 \in [m]$ and $l \in [k]$.

When the $m+k$ equations are summed we receive:

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

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

which gives us the updates

$$T_{j_1,j_2} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j_1,0), y_t = (j_2,0)\right)}{\sum_{t\in 2\ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j_1,0)\right)} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}\left(y_{t-1} = (j_1,0), y_t = (j_2,0)|x_{1:L}\right)}{\sum_{t\in 2\ldots L} P_{\theta'}\left(y_{t-1} = (j_1,0)|x_{1:L}\right)} \tag{13}$$

$$G_{j,l} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j,0), y_t = (j,l)\right)}{\sum_{t\in 2\ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j,0)\right)} = \frac{\sum_{t\in 2\ldots L} P_{\theta'}\left(y_{t-1} = (j,0), y_t = (j,l)|x_{1:L}\right)}{\sum_{t\in 2\ldots L} P_{\theta'}\left(y_{t-1} = (j,0)|x_{1:L}\right)} \tag{14}$$

E-step

Preceding the M-step where we update components of θ by (11), (12), (13) and (14), we will calculate the three probability terms inside them in the E-step, denoted as γ, ξ and η:

$$\gamma_{j,t} = P_{\theta'}(y_t = (j,0)|x_{1:L}) \tag{15}$$

$$\xi_{j_1,j_2,t} = P_{\theta'}(y_{t-1} = (j_1,0), y_t = (j_2,0)|x_{1:L}) \tag{16}$$

$$\eta_{j,l,t} = P_{\theta'}(y_{t-1} = (j,0), y_t = (j,l)|x_{1:L}) \tag{17}$$

For the calculation of these probabilities, we first need to calculate the forward and backward probabilities output from an HOP-HMM adjusted Forward-Backward algorithm. In this HOP-Forward-Backward algorithm, we will only build the probabilities for being in background-states since the TF-states probabilities are not needed in the later parts of the E-step.

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

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

The adjustments for the forward and backward algorithm are straight forward, as the summation is composed of two parts. We calculate α of size $m \times L$, iterating over $t = 1,2,...,L$ as following:

Algorithm 3 HOP Forward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

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

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

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

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

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

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, 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 used 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. Also, when summing the TF-state transition part, PWMs W_l with length equal or bigger than $t+1$ include out-of-sequence TFBS and are not part of the summation.

For β of size $m \times L$, we iterating over $t = L, L-1, ..., 1$ as following:

Algorithm 4 HOP Backward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

$\beta_{1:m,L} = 1$

$for\, t = [L-1, ..., 1]:$

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

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

Note that when $t > L - |W_l|$, there are missing observable variables to fully calculate the TF-state transition. In these situations this contribution of these component to the summation is zero, meaning our HOP-HMM as the behavior of avoiding a transition into a TF-state when the PWM is too long to fit into the sequence X length.

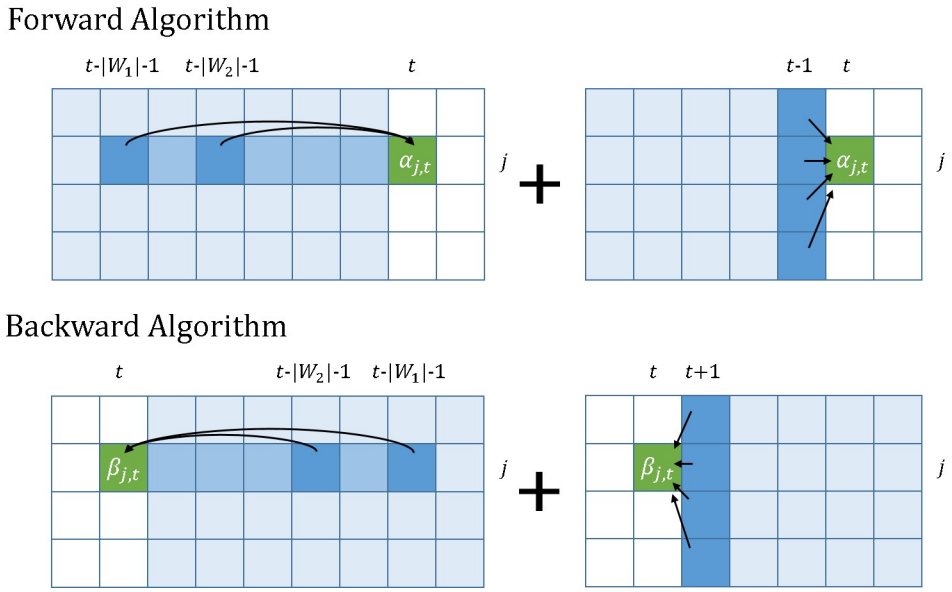


Figure 15: In HOP-HMM, the Forward-Backward algorithm dynamic tables α and β cells are filled from both the adjacent background-states transitions and the background-states preceding or proceeding the motifs emitted by the TF-states.

We will now explain why the described dynamic calculation result with $\alpha_{j,t} = P(y_t = (j,0), x_{1:t})$ and $\beta_{j,t} = P(x_{t+1:L} | y_t = (j,0))$, starting with the forward probabilities α. From the law of total probability, the probability $\alpha_{j,t}$ is the sum of probabilities of all the possible transition that ended in the background-state (j,0):

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

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

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

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

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

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

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

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

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

$$= \alpha_{j,t-|W_l|-1} \cdot G_{j,l} \cdot L_{W_l}\left(x_{t-|W_l|}, ..., x_{t-1}\right) \cdot E_{j,x_{t-o+1},...,x_t}$$

This process is similar to the background-state transition. Using the chain rule:

$$P\left(y_{t-1} = (j',0), y_t = (j,0), x_{1:t}\right) =$$

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

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

$$= \alpha_{j',t-1} \cdot T_{j',j} \cdot E_{j,x_{t-o+1},...,x_t}$$

For the backward probabilities β, the explanation is similar. The main difference between the regular HMM backward probability is the condition on the $o-1$ preceding observable variables $x_{t-o+2:t}$, which are necessary for the background-state emission is conditional on them.

Using the law of total probability:

$$P\left(x_{t+1:L} | y_t = (j,0), x_{t-o+2:t}\right) =$$

$$= \underbrace{\sum_{j' \in [m]} P\left(y_{t+1} = (j',0), x_{t+1:L} | y_t = (j,0), x_{t-o+2:t}\right)}_{\text{background-state transition}} + \underbrace{\sum_{l \in [k]} P\left(y_{t+1:t+|W_l|} = (j,l), y_{t+|W_l|+1} = (j,0), x_{t+1:L} | y_t = (j,0)\right)}_{\text{TF-state transition}}$$

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

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

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

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

$$=P\left(x_{t+2:L}|y_{t+1}=(j',0),x_{t-o+3:t+1}\right)\cdot P\left(x_{t+1}|y_{t+1}=(j',0),x_{t-o+2:t}\right)\cdot P\left(y_{t+1}=(j',0)|y_t=(j,0)\right)=$$

$$=\beta_{j',t+1}\cdot E_{j',x_{t-o+2},\ldots,x_{t+1}}\cdot T_{j,j'}$$

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

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

$$P\left(x_{t+|W_l|+2:L}|x_{t+1:t+|W_l|+1},y_{t+1:t+|W_l|}=(j,l),y_{t+|W_l|+1}=(j,0),y_t=(j,0)\right)\cdot$$

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

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

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

$$=P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0)\right)\cdot P\left(x_{t+|W_l|+1}|y_{t+|W_l|+1}=(j,0)\right)\cdot$$

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

$$=\beta_{j,t+|W_l|+1}\cdot E_{j,x_{t-o+|W_l|+1},\ldots,x_{t+|W_l|+1}}\cdot L_{W_l}\left(x_{t+1},\ldots,x_{t+|W_l|}\right)\cdot G_{j,l}$$

Using the forward and the backward probability matrices α and β, we can calculate the auxiliary probabilities (15), (16) and (17). The first probability that will help us for that is ψ, a matrix of size $m\times k\times L$:

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

$$=P\left(y_t=(j,0),y_{t+1}=(j,l),y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)\cdot P\left(x_{t+|W_l|+2:L}|y_{t+|W_l|+1}=(j,0),x_{1:t+|W_l|+1}\right)=$$

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

$$\cdot P\left(x_{t+|W_l|+1} | y_{t+|W_l|+1} = (j,0)\right) \cdot P\left(x_{t+|W_l|+2:L} | y_{t+|W_l|+1} = (j,0), x_{t+|W_l|-o+3:t+|W_l|+1}\right) =$$

$$= \alpha_{j,t} \cdot G_{j,l} \cdot L_{W_l,}\left(x_{t+1}, ..., x_{t+|W_l|}\right) \cdot E_{j, x_{t+|W_l|-o+2}, ..., x_{t+|W_l|+1}} \cdot \beta_{j,t+|W_l|+1}$$

The second probability is likelihood of the observed sequence $x_{1:L}$:

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

Now we can calculate probability (15) of the background-state at a given position given the sequence X, denoted as γ of size $m \times L$:

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

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

The probability (16) is the background-state to background-state transition given the sequence X, denoted as ξ of size $m \times m \times L$:

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

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

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

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

Finally, the probability (17) is the background-state to background-state transition given the sequence X, denoted as ψ of size $m \times k \times L$:

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

Now with (15), (16) and (17) at hand, we can complete the M-step and update θ by assigning the updates of (11), (12), (13) and (14).

The Baum-Welch algorithm adaptation for HOP-HMM, as described in this section:

Algorithm 5 HOP Baum-Welch

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

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

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

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

If θ converged, break EM for loop

return θ

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

Sequence States Inference

Acquiring the maximal likelihood θ opens the door to several wanted inferences given a sequence $x_{1:L}$:

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

γ (15) and η (17) can be used to solve the inference 1 for HOP-HMM. We aim to maximize here a posterior probability in a specific position:

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

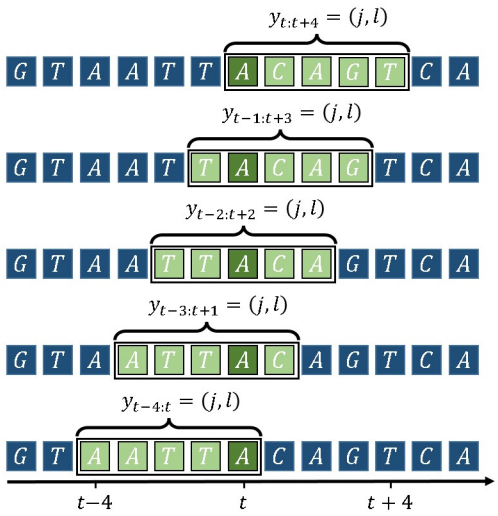
In regular HMM, we can approximate this by taking the max of the posterior probability held in γ (9) built by a θ that we learned with the Baum-Welch algorithm. In HOP-HMM γ (15) is not sufficient since it holds only the probability of background-state $P_\theta(y_t = (j,0)|x_{1:L})$. To calculate the posterior probability for TF-states, $P_\theta(y_t = (j,l)|x_{1:L})$ where $l > 0$ we sum all options of a TF-state (j,l) that cover position t as described in (16).

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

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

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

$$\hat{y}_t = \underset{j\in[m],l\in[k]\cup\{0\}}{argmax} \gamma_{j,t} \cup \sum_{i\in[|W_l|]} \eta_{t-i+1,j,l} \tag{18}$$



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

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 y_t. In this example, W_l is a PWM of length 5, therefore it has 5 different possible positions that include y_t that are summed, marked in light green.

Inference 2 aims for reaching the most likely hidden sequence:

$$y_{1:L}^* = argmax_{y_{1:L}} P(y_{1:L}|x_{1:L}) \tag{19}$$

The main difference between inference 1 is the consideration to the dependency between adjacent states. In inference 1, for example, two adjacent positions may been individually inferred states which the transition probability between them equals 0. Even though each hidden state maximizes the likelihood at its position, as a sequence when accounting for the transitions the result might not be the same states.

HOP-Viterbi Algorithm

In HMM, deriving the maximal likelihood hidden sequence of (19) is done by the Viterbi algorithm, named after Andrew Viterbi who proposed it in Viterbi (1967). Viterbi algorithm resembles the Forward algorithm, with the two main differences:

1. Maximization replaces the summation over the possible transitions.
2. Traces of the maximal value chosen in the dynamic filling are kept in V^2, which used to back-trace the chosen state path at the end.

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 a HOP-Viterbi algorithm, in two manners:

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

Algorithm 7 HOP-Viterbi Algorithm

Input:

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

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

Algorithm:

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

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

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

$fort = [2,...,L]:$

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

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

$B = \left\{V_{j,t-|W_l|-1} \cdot G_{j,l} \cdot L_{W_l}\left(x_{t-|W_l|},...,x_{t-1}\right) \cdot E_{j,x_{t-o+1},...,x_t} | l \in [k]\right\}$ *# background-state to TF-state*

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

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

$\hat{y}_L = \left(argmax_j V^1_{j,L}, 0\right)$ *# 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:

$y_{t-|W_l|:t-1} = (j,l)$

$y_{t-|W_l|-1} = y_t$

$t = t - |W_l| - 1$

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

Using the Viterbi state path, we can make a simplistic classifications of short DNA sequences by their dominant state. This simple classification made by choosing the most repeating state in the estimated Viterbi state path $y_{1:L}$:

$$y_{class} = mode_{t \in [L]} y_t$$

Results

For evaluating HOP-HMM, we first measure its capabilities on synthetic DNA data that was created in a controlled way. Afterwards, we see how does a HOP-HMM classify real human DNA sequences. The evaluation process on synthetic data is done by the following steps (see figure 17):

1. We generate parameters for a HOP-HMM θ, which are treated as the true θ. θ is sampled in the following way:
   (a) Each cell T is sampled from the uniform distribution
   $$T_{i,j} \sim U\left(minT_{i,j}, maxT_{i,j}\right) \tag{20}$$
   (b) Each cell $G_{i,j}$ is sampled from a uniform and a Bernoulli distribution
   $$G_{i,j} \sim U\left(minG, noiseG\right) + 1_{(i,0) \in Reg} \cdot Bern\left(\frac{k}{m}\right) \cdot maxG \tag{21}$$

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 high probability of transitioning into TF-state. The rest of the background-states will have low probability to create TFBS, aiming to model the non-regulatory regions of the DNA with sparse TFBSs. In our experiments ENH contained all but one state: $(m,0)$ meaning only one background-state had almost no TFBS and the rest $m-1$ background-states did have TFBSs.

(c) After being sampled, T and G are divided element-wise by their rows sum so their rows together become distributions:

$$T_{i,j} = \frac{T_{i,j}}{\sum_{j'\in[m]} T_{i,j'} + \sum_{j'\in[k]} G_{i,j'}} \qquad G_{i,j} = \frac{G_{i,j}}{\sum_{j'\in[m]} T_{i,j'} + \sum_{j'\in[k]} G_{i,j'}} \tag{22}$$

(d) E is 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, similar to (c):

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

(e) The start state distribution π is non-random, and set so the first states are always one of the non-enhancer background-states:

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

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

3. From the DNA sequences of X train, we train a $\hat{\theta}$ with the HOP Baum-Welch algorithm.

4. Using the trained parameters $\hat{\theta}$, we estimate $\hat{Y}$ test from X test and $\hat{Y}$ train from X train by the HOP-Viterbi algorithm. We also calculate the posterior probability of $P_{\hat{\theta}}(y_t|x_{1:L})$ from X test and X train. These results are then compared to the real Y test and Y train for accuracy measuring.

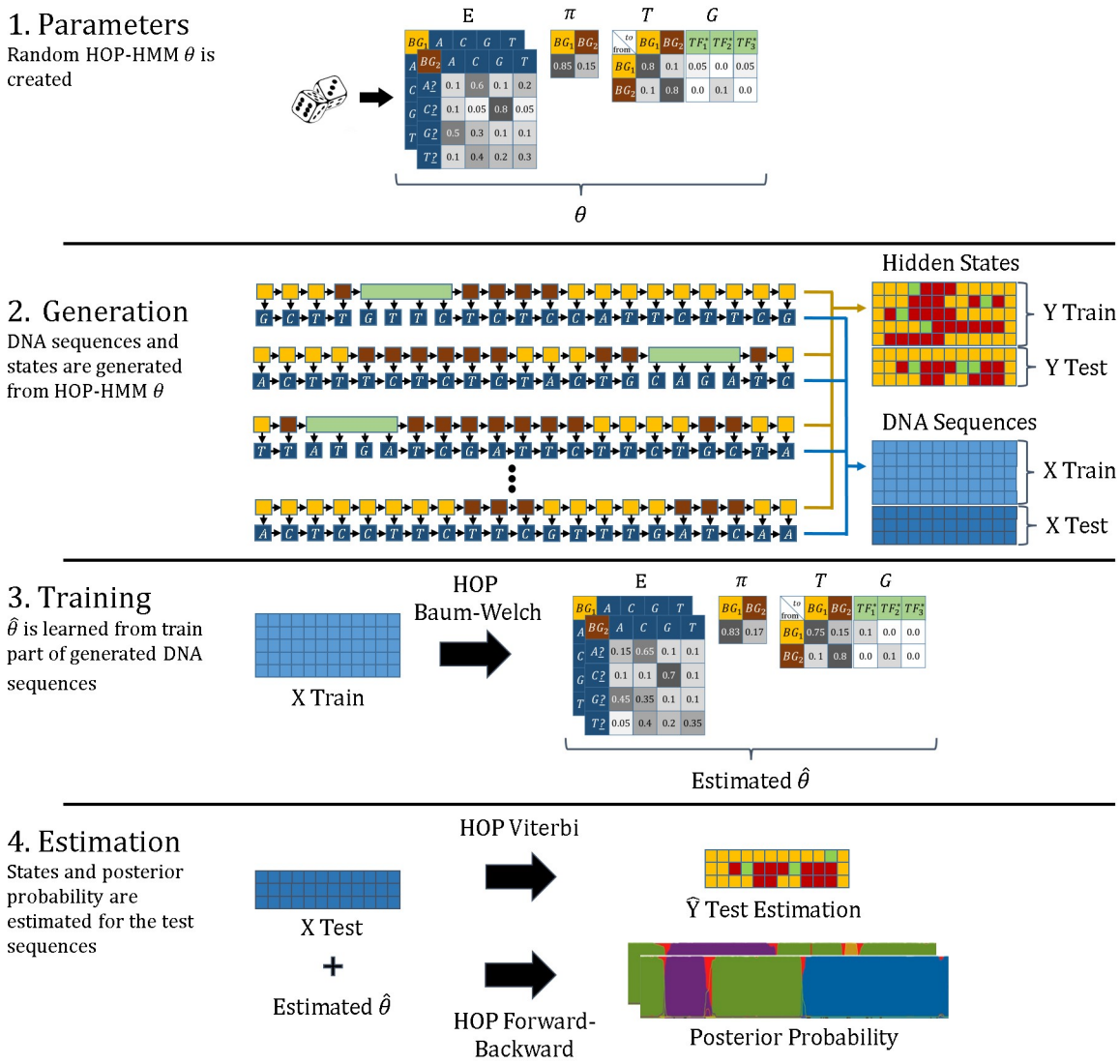


Figure 17: Workflow of the evaluation process. A θ is sampled and a HOP-HMM model is created with which several fixed-length sequences are generated. A new model $\hat{\theta}$ is then fitted to the train section of the observed sequences, via HOP-Baum-Welch algorithm. With $\hat{\theta}$, a hidden sequence is then estimated by the HOP-Viterbi algorithm, and a posterior probability estimation is calculated by (18).

The Baum-Welch algorithm is guaranteed to increase the likelihood in each step, however it is not guaranteed to converge to the optimal θ^* (Rabiner (1989)) as there is no known analytical way for reaching it. As a consequence, Baum-Welch algorithm converges into a local maximum $\hat{\theta}$ which could be a relatively low likelihood estimation, depending on the initialization point of the first θ. The local maximum convergence issue is addressed in two ways:

1. We use regularization for faster and to a better $\hat{\theta}$ convergence (see figure 19). Following each M-step update we draw the background-states transition probabilities T to remain between $maxT$ and $minT$ matrices from (20):

- 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 are divided by their row sum so their rows together remain a distribution as in (22)

2. Since Baum-Welch seek local maximum, running it multiple times with different initializations will cause convergence into 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). This is important since on real observed sequences, only the estimated $\hat{\theta}$ likelihood is known while the true θ is unknown. This correlation implies that for an estimation $\hat{\theta}$ that closer to the true θ, one should redo several EM runs and choose the $\hat{\theta}$ with the highest likelihood.

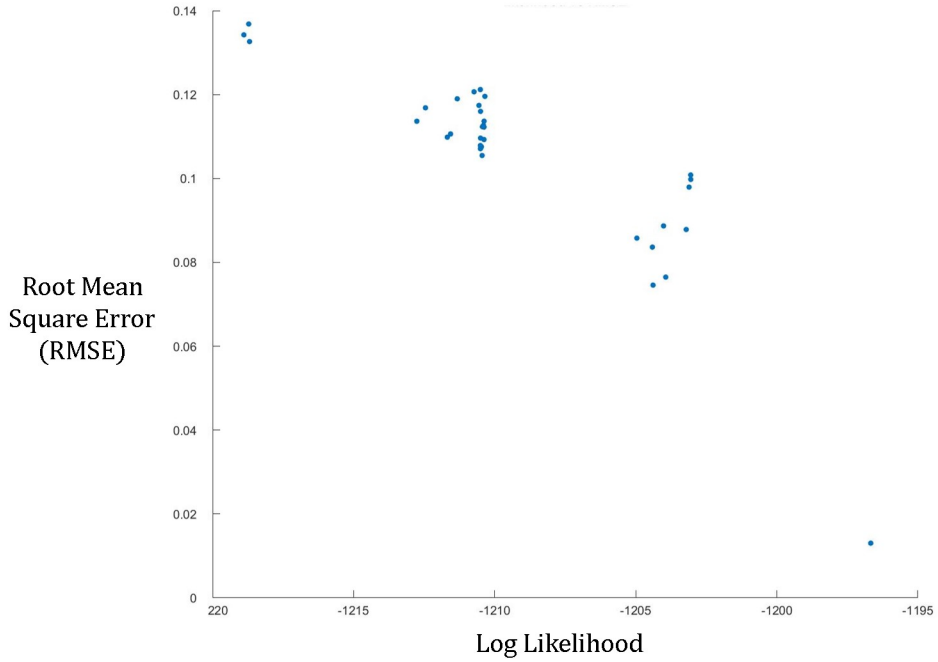


Figure 18: Over multiple runs of HOP-Baum-Welch, higher the sequences likelihood for the estimated θ resulted in lower errors compared to the true θ.

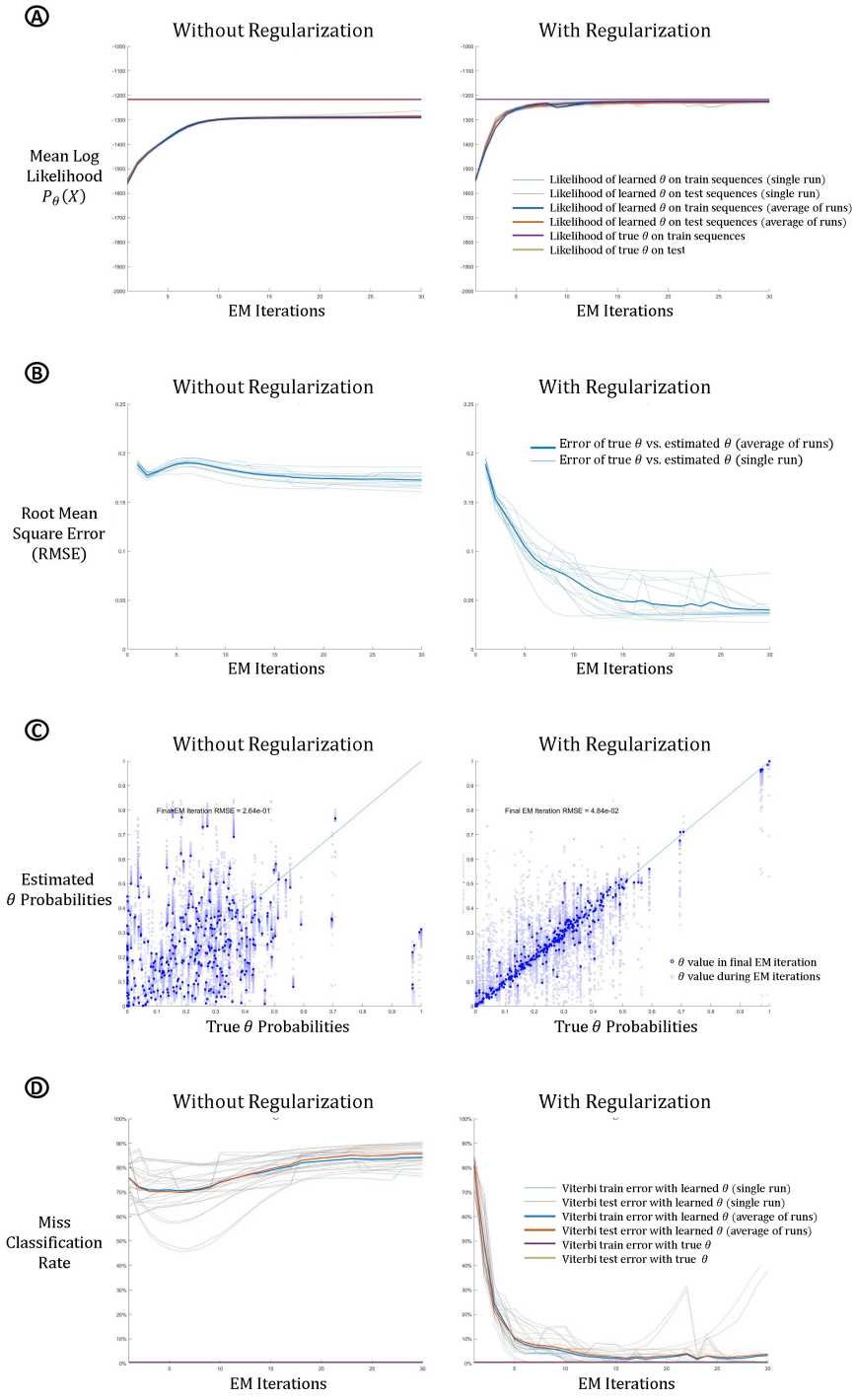


Figure 19: **A)** The EM iterations draws the estimated θ values mostly closer to the values of the true θ. **B)** The error between the true and estimated θ decrease, and after a few iterations converge to the same path regardless of the initialization. **C)** During the EM iterations, the learned θ yields a more accurate Viterbi estimation of the hidden states. Note that not even the true θ could produce Viterbi paths that is a perfect match to the true hidden sequences. **D)** The mean log likelihood of the sequences increases during the EM iterations. The experiment was done on 500 synthetic sequences (85% train, 15% test), 1000 long. The trained model had 6 hidden background-states with emission order of 2, each background-state had 25 TF-states.

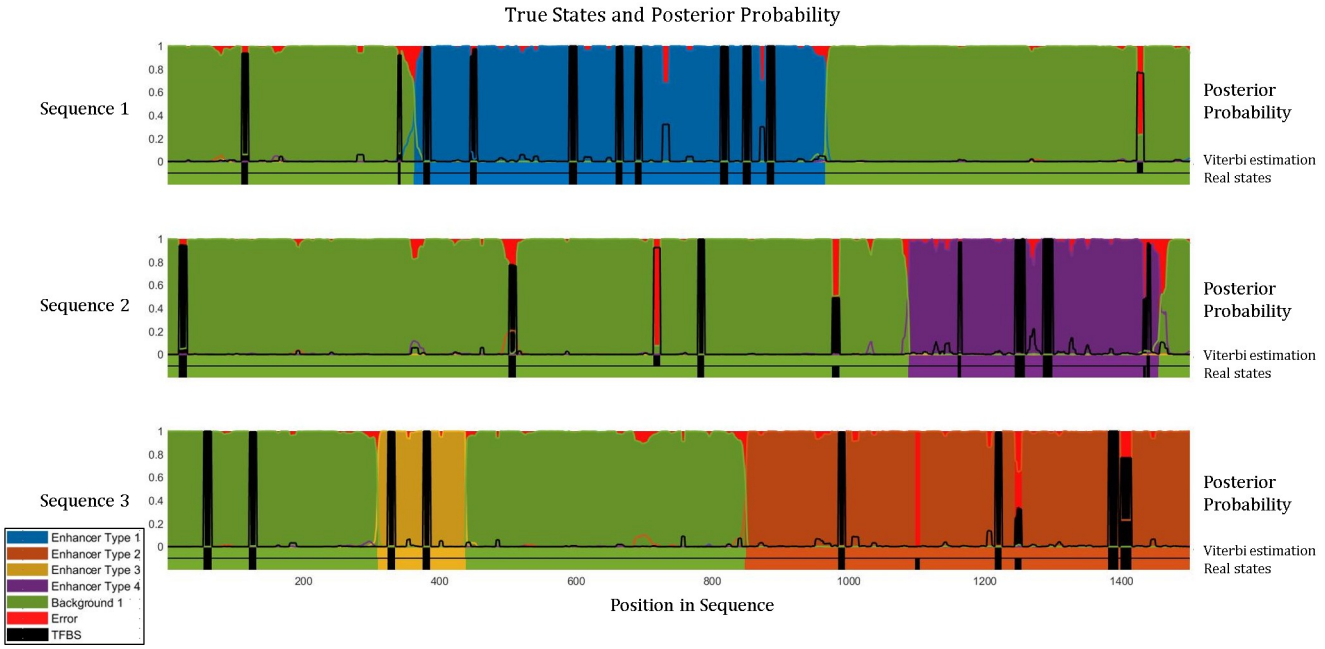


Figure 20: Posterior probability of sequences, estimated by a trained HOP-HMM $\hat{\theta}$ on test sequences that were synthetically generated by a HOP-HMM θ. At the bottom of each posterior probability, there are the Viterbi hidden path by $\hat{\theta}$ and the true hidden states of each sequence. The black TFBS is the sum of all the probabilities of being in any of the TF-states.

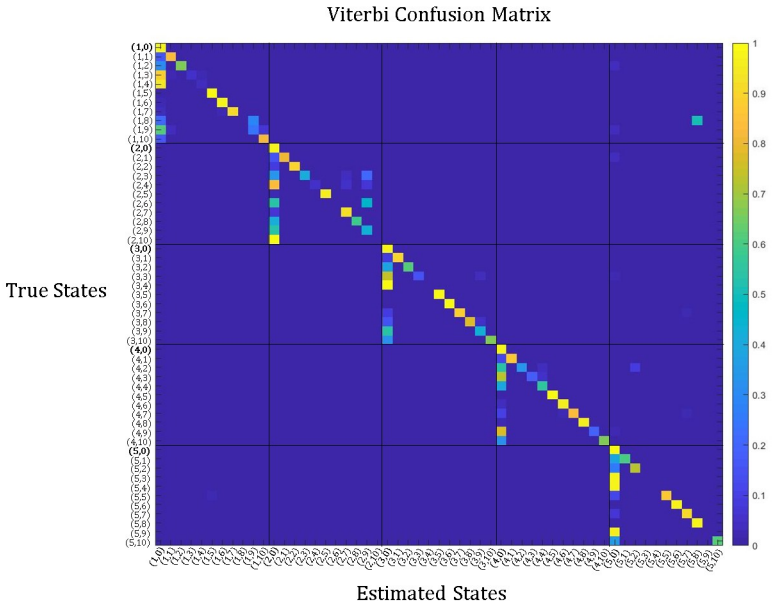


Figure 21: Confusion matrix of true and estimated states by the Viterbi algorithm of HOP-HMM synthetic sequences. Rows are normalized so their sum is equal to 1. The majority of prediction are in the background-states $(1,0)$, $(2,0)$, $(3,0)$, $(4,0)$ and $(5,0)$, where TF-states are sometimes misclassified as their background-state state.

For testing of HOP-HMM on human genetic data, we manipulated the used the Roadmap project Bed files with BEDTools (Quinlan and Hall (2010)). We created a dataset of enhancer sequences around the intersection of DNase-I, H3K27ac and H3K4me1 peaks, while avoiding peaks of H3K27me3 and H3K4me3 and sequences within 5000bp from known genes. Sequences were chosen 5000bp long sequences centered around their DNase-I peaks.

We wanted to evaluate if HOP-HMM can distinguish and detect enhancers active in two human tissues. For this we've built a set of 4000 bp fixed length sequences from the HG19 in positions which were labeled as enhancers based on epigenetic data collected by Roadmap project from 57 tissues. Out of these enhancers, we chose only sequences of tissue-specific enhancers in one of two types of tissues, and not in the rest of the 57 tissues. We've added sequences with no known role, from random locations in the genome distal from genes or enhancers as background sequences.

A HOP-HMM is trained by the HOP-Baum-Welch algorithm on the collected sequences. The trained model is then used to produce a Viterbi estimated hidden-states sequence and posterior probability, which can then be compared to the epigenetic tracks.

For the set of PWMs used by the TF-states of the HOP-HMM, we used JASPAR dataset of 519 vertebrates PWMs, out of which we selected 50 PWMs for a practical run-time. The selected PWMs are chosen by 3 methods, each method responsible for a third of the 50:

- PWMs of TFs which were relatively expressed in one tissue compared to the other, according to the Roadmap RNA-seq data. This method does not depend on the sequences themselves, but on epigenetic properties of the tissues.
- PWMs which are abundant in the sequences, i.e. PWMs with the highest mean likelihood to attach to the sequences. The likelihood of PWM W to bind to a sequence x is calculated as described mean is the average of 3 highest likelihood TFBS as described in figure 3. Note that here we use the PWM form as in (1) and not the PPM form for comparison between PWM likelihoods.
- PWMs that had strong presence in sequences from one tissue compared to the other. Specifically, the PWMs with sequence binding likelihood (as defined in the previous method) that can best distinguish between sequences from one tissue and the rest of the sequences in terms of AUC-ROC.

In our experiment, some sequences resulted posterior probability had good resemblance to the DNase-seq track, causing a good overlap between the Viterbi-path and the ChromeHMM classifications (see figure 22), though this similarity was not always occurring.

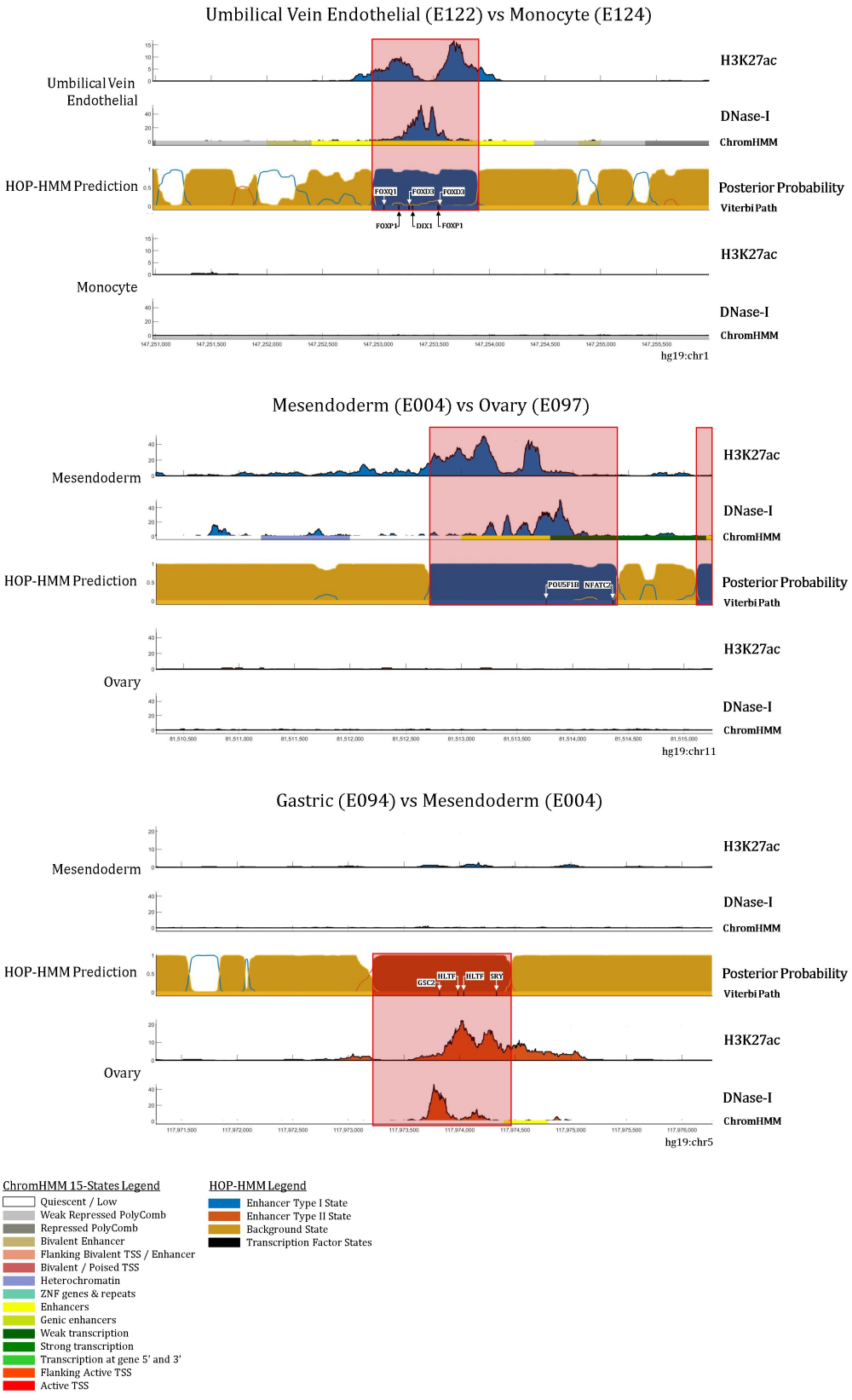


Figure 22: Examples of HOP-HMM classification of 5000pb long tissue specific enhancer sequences from the human genome. Each of the 3 graphs is the output of a different HOP-HMM with three background-states (two enhancer states and one non-enhancer state) and 50 TF-states, each was trained on enhancers from the two tissues. The H3K27ac and DNase-I measurements are in $-log_{10}(p-value)$ units.

Discussion and Conclusions

In this work I have aimed to develop a generalized HMM, HOP-HMM, tailored for the enhancer structure. I developed the mathematical adjustments to the different parts of the EM algorithm and provided and reasoning for the correctness of the inferring of the altered model from data. For evaluation on real and synthetic data as described in the results, I implemented the model and the algorithm in Matlab code. During the algorithm implementation, I overcame a few difficulties that origin 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 to sequences into batches for holding and manipulating the large η (17) matrix in memory. The implementation also includes the generation code for DNA sequences from a randomly selected HOP-HMM to which a different model can be fitted and compared. Naturally, in the synthetic data experiment, the more generated data is used to fit the model, the better the fitting performs. During the evaluation of the inference EM algorithm, many runs tend to overshoot the inter-states transition probability, resulting in a tendency for irregular Viterbi paths with frequent state changes. This resembles the known issue of HMM parameter overfitting on small training data. I therefore introduced a regularization technique that had a significant positive impact on the convergence rate and solution quality. Overall, the generated DNA sequences experiment results were positive and are evidence for the ability of the algorithm to train successfully on DNA sequences created under the HOP-HMM assumptions.

For applying the EM algorithm on real data, I designed a simplistic challenge of correctly recognizing enhancers where their true locations are known from epigenetic experiments. This challenge first required building a dataset of sequences containing tissue-specific enhancers from two tissues together with wide margins, and non-regulatory "background" sequences. When choosing the tissue-specific enhancers from the epigenetic data provided by the Roadmap project, the noisiness of the data should be considered. After some trail and error, I chose the somewhat arbitrary cutoff of the top 40% strongest DNase-seq peaks, which yielded enough sequences (around 500 sequences per tissue on average) with distinguishable distributions between the tissues.

Though many real DNA sequences were correctly classified, some tissues caused the EM algorithm to consistently converge to parameters which gave mostly wrong Viterbi path classifications. This could stem from several reasons: wrong PWMs selection as hyperparameters, too small or noisy dataset building from the Roadmap data and a more complex structure than assumed by the HOP-HMM hypothesis. In further research, usage of updated data with cleaner more tissue-diverse experiments would likely provide better results. As a computational approach to improve the generalized HMM used in this work is to introduce learning of the PWMs preceding or during the EM iterations, or to replace the PWMs emission entirely by a different TFBS modeling method.

Appendix: Source Code

The code for this research was written in Matlab, and can be found in `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

The prominent code files in the project:

- **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, process 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-state to TF-state transition, marked as *noiseG* in (21)

- **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, doESharing, doGTBound);

  theta = genTheta(params, false, false);
  ```

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

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

  ```
  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 sequence and posterior probabilities similar to figure 20 and a confusion matrix similar to figure 21.

  ```
  mainPosterior();
  ```

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

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

  ```
  mainDecErrorPlot();
  ```

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

  The function actually trains the HOP-HMM model 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 that are allowed in a run and `repeat` is the number of different runs with different initializations that are tried.

References

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.

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.

Buenrostro, J. D., Giresi, P. G., Zaba, L. C., Chang, H. Y., & Greenleaf, W. J. (2013). Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. Nature methods, 10(12), 1213.

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

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.

Diehl, A. D., Meehan, T. F., Bradford, Y. M., Brush, M. H., Dahdul, W. M., Dougall, D. S., ... & Van Slyke, C. E. (2016). The Cell Ontology 2016: enhanced content, modularization, and ontology interoperability. Journal of biomedical semantics, 7(1), 44.

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.

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.

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.

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.

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.

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 high-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.

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

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.

Staden, R. (1984). Computer methods to locate signals in nucleic acid sequences.

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.

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.

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., 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.