High-Order Generalized Hidden Markov Model for DNA Regulatory Sequences Classification

May 23, 2019

Background

The Genome

The genome of every organism contains the inherited information that defines its complex structure and function. The genome is built out of Deoxyribonucleic acid (DNA) molecule, that is a built out of two chains of nucleotides units that form a double helix shape. Each nucleotide is built our of 4 different types bases: cytosine, guanine, adenine or thymine or in short A,C,G and T. The nucleotides are organized in pairs called base pairs where each of the paired nucleotides are complimentary to each other and provide redundancy.

Proteins are macromolecoles, which carry various roles and functions within organisms. They are built out of 20 different amino acids, which order and structure is encoded inside genetic segments in the genome called genes. Through the transcription and 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 RNA molecules are translated into a sequence of amino acids that constitute a protein.

Genes Gene sequences are built out fragmented introns and exons, where only the exons becomes the RNA molecules that translates into proteins while the introns are spliced away beforehand. Although the exons alone hold the recipe for the construction of the organism's proteins, the complexity of the organism is not a product of their number or their length. For example, the humans and Caenorhabditis elegans roundworms both have about 19,000 genes (Ezkurdia et al. 2014; The C. elegans Sequencing Consortium 1998), with roughly the same total exon length and number, although the human body is vastly more diverse and complex. The source for the organisms complexity differences is attributed to the gene regulation mechanism. The human genome is 3.23 Gb long, and it is estimated that gene regulation involves 10-20% of it (Pennacchio et al. 2015), compared to 1% that are exon regions (Ng et al. 2009).

Enhancers Enhancers are are non-coding regulatory DNA sequences that 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, and their length are usually between 100-1000 bp. When activated, the DNA folding draws the enhancer spatially closer to another type of regulatory element called promoter, resulting in the translation of a gene adjacent to the promoter (see figure 1). The enhancer's target gene is the expressed gene from this activation process. It can be located up to a megabase upstream or downstream from their activating enhancer (May et al. 2011), and are orientation independent to it. Moreover, the gene-enhancer connection is not exclusive, and the common case is that each enhancer has several target genes and vice versa (Fishilevich et al. 2017).

Gene Transcription

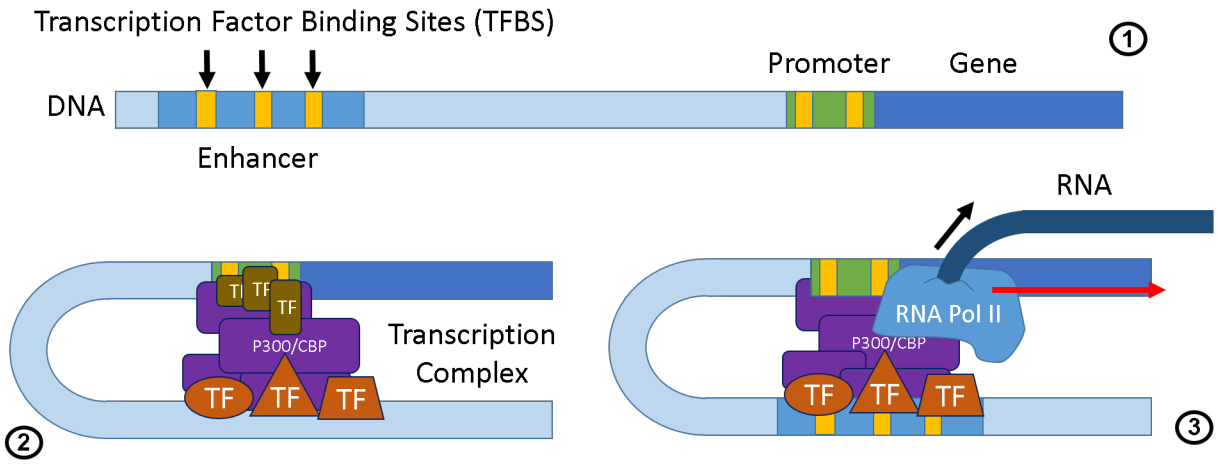


Figure 1: 1) An enhancer and its distal target gene. 2) The DNA folds and the attached with transcription factors draw other co-factor proteins that together form the transcription complex. 3) The RNA Polymerase II is recruited and while moving along the gene it generates a new RNA molecule that is transcribed off the gene sequence.

Transcription Factor Binding Transcription factors (TF) are proteins that bind to the DNA, and together with other co-factor proteins initiate the gene transcription process. TFs tend to bind to certain transcription factor binding sites (TFBS), which are motifs of nucleotides on the DNA with average length of 12 bp in humans (Kulakovskiy et al. 2011) that are conserved between species (Doniger et al. 2005). On genome-wide association studies (GWAS) done with ChIP-seq method, different TFs have different distributions of TFBS they are observed attached to (Khan et al. JASPAR 2018; Gheorghe et al. 2018).

Both enhancers and promoters contain TFBSs that are critical for the their correct regulatory operation. Multiple studies have shown that genetic alternations in TFBS can affect the expression of the regulated gene and are a major cause of different human diseases (Kreimer et al 2017; Miguel-Escalada 2015; Soldner 2016; Smemo S. 2012; Benko 2009; Emison, 2005; Lettice 2003). From the sequence aspect, enhancers and promoters have a similar structure of a background nucleotide sequence with distribution different from other part of the genome, with TFBS motifs tiled inside this background sequence.

The enrichment of TFBS is a good predictor for the location of promoter and enhancer regulatory regions and the type of cells they will be active in. Folding of DNA allows the enhancer-promoter interactions, in which the TFs take major part. Once bounded to the DNA, the TFs recruit other cofactor proteins to them, and together they form a transcription preinitiation complex (PIC), 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 role of transcribing the adjacent gene. it opens the double stranded DNA, so that one strand of nucleotides is exposed and becomes a template for RNA synthesis.

PWMs Generating a compact model for estimating the binding potential of a DNA sequence to a TF, i.e. $P(x_{1:n}|binding)$, is not trivial as might seem on first look. The peaks of the ChIP-seq data are used as the ground truth of TF binding locations, from which the model is built. Position weight matrix (PWM) is a commonly used simplistic method 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:

$$P(x_{1:n}|binding) = \prod_{i \in [n]} P(x_i|binding)$$

Where n is the size of relevant sequence. The size of the sequence that is affected by the binding event is derived from the physical characteristics of the TF.

$P(x_i|binding)$ is estimated by counting the nucleotides frequency in every position of the observed binding sites, which are the ChIP-seq peaks. For a motif of length J, this probability estimation is stored in a PWM matrix W as followed: $W_{i,j} = \frac{1}{N}\sum_{k\in N}\mathbf{1}(X_{i,k} = j)$ where $i \in [J]$ the position in the motif and $j \in [4]$ the nucleotide index of A,C,G and T.

From a generative model point of view, the sequence is generated by a TFBS motifs emission system. For this needs, the log of the matrix often comes handy for calculation of $log\left(L(W; x_{1:n})\right)$, the log of the probability that a motif was generated by a PWM W. This calculation is done be a convolution of $log(W)$ on a one-hot encoding of the sequence.

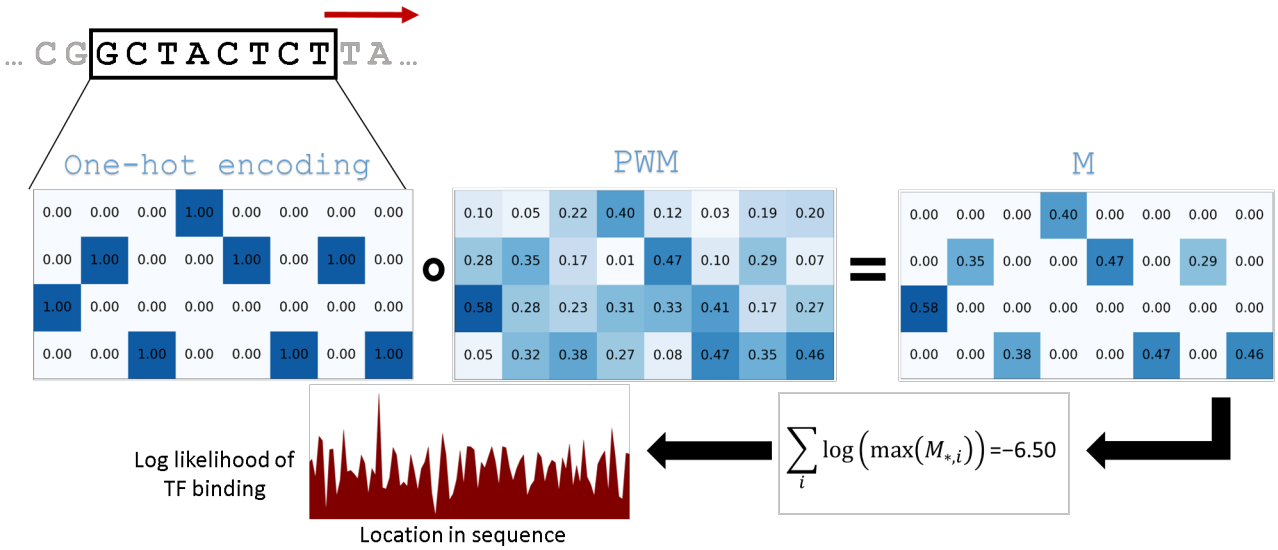


Figure 2: a sub-sequence out of the DNA is represented in a one-hot encoding, then entry-wise multiplied with the PWM. Then, the sum of the logs of the maximal values of each column in the result matrix is calculated, which is the log likelihood of the TF binding to the sub-sequence. This log likelihood is calculated for each location in the sequence, where location with high values indicate high likelihood of TF binding.

Inter TFBS Sequences and Conservation Conserved non-coding elements (CNE) reside in clusters, usually with low gene density but with vicinity to genes. Typically, CNE are structured in arrays called GRB, with a mean length of 1.4 Mb (Dong et al. 2009). The correlation between conservation of non-coding region and enhancer functionality is not strong. Some verified enhancers are weakly or not conserved between distant species (Friedli et al. 2010; Rosin et al. 2013; Taher 2011; Lindblad-Toh 2011) and some highly conserved areas in the mouse genome are not associated to regulatory activity and their deletion and yielded 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 enhancers activity in mice (Visel et al., 2007). The reason for such ultra-conservation of 200 bp sequences when the TFBS is only 4-8 bp long is unclear. It is possible that these conserved sequences are actually long assembly of overlapping TFBS or that the enhancer has another function as a eRNA, that the exact nature of its mechanisms is no understood (Haeussler et al. 2011, Andersson et al. 2014).

Epigenetics

Almost all cells in every organism contain its genome, but only part of genome is active in any specific cell. Cells of different types and in different operation modes differ by gene expression patterns. The reason for that lies in regulation components that are outside of the genomic sequence. The location and presence of TFBS, background nucleotides distribution and other sequence related properties are not enough to explain regulatory role of regions in the genome.

Several epigenetic features (which do not involve the nucleotides sequence directly) correlate with enhancer regions in the genome:

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

These properties and mechanisms have measurable features that lie on top of the genome. Their combination is the main source of identification for enhancer regions in the genome. Each cell has its own epigenetic features, in a binaric form, e.g. a specific part of the genome can be either accessible, or not. When several similar cells from the same tissue sample are measured, a frequency or count of the feature is measured per DNA loci, and generates epigenetic data. The epigenetic data is commonly used as the ground truth indication for enhancer sequences, as done for the human genome in the ENCODE project.

[TODO: fix ENCODE cite above]

Accessibility In eukaryotes, the DNA is packed around a structure of 8 histone proteins, together forming a nucleosome core. The location of the nucleosome binding is not random over the DNA sequence, but has a tendency for specific DNA binding sites (Cutter et al. 2015). DNA that is wrapped around a nucleosome has a lesser probability to interact with proteins, as it is physically inaccessible. Both the enhancer, the promoter and the gene need to be accessible for a successful transcription.

Since the scenario of TF binding on an enhancer requires an accessible DNA region, I hypersensative sites are used for detecting a potential DNA cleavages that have the potential of being regulatory elements, in usually a better resolution than histone marks.

Histone Marks Chromatin modifications signatures, also called histone marks, are predictive of enhancer position and activity status and can be assessed (Visel et al. 2009; Firbi et al. 2010; Fernandez et al. 2012). The histone marks are considered to contain a certain "histone code" which encode complex information, additionally to the DNA, regarding the transcription regulation and other aspects. Comparing to other epigenetic information, and especially DNA methylation (Przybilla et al. 2012), chromatin modifications have a short time-scale of seconds or hours (Hayashi-Takanaka et al., 2011), hence they are considered part of the dynamic changes of the cell's modes.

H3K4me1 and H3K27ac are among the predominant histone marks of active enhancers, where H3K4me1 are enriched on transcribed genes and enhancers prior to activation (calo et al. 2013), and is thought to precede the H3K27ac modification (Creyghton et al., 2010; Rada-Iglesias et al., 2011; Zentner et al., 2011) which is known to occur during the activation. Other histone marks that are present on active enhancers and are used for their detection are H3K9ac (Ernst et al., 2011; Karmodiya et al., 2012; Krebs et al., 2011; Zentner et al., 2011) and H3K18ac (Jin et al., 2011). Even though H3K27ac have been identified as an important mark for distinguishing active enhancers from poised enhancers (Creyghton et al. 2010), it is not enough as its own since when present alongside H3K4me3 it is an indication for active promoters [Heintzman et al., 2007]. In contrast, H3K27ac absence and H3K4me1and H3K27me3 enrichment are typical for poised enhancers (Creyghton et al, 2010).

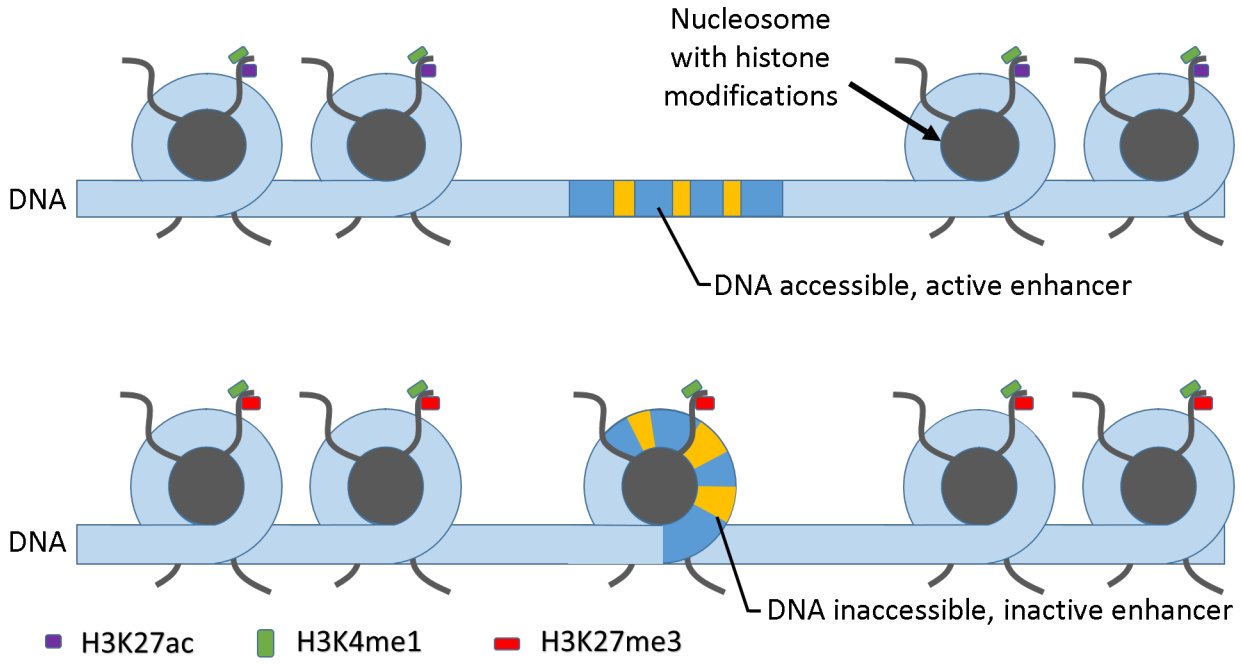


Figure 3: The accessibility of the enhancer's sequence and its surrounding histone marks are connected to its regulatory activity state. On the upper part an active enhancer sequence that is accessible for protein interaction needed for transcription, where as on the lower part an inactive enhancer is inaccessible since it is wrapped around a nucleosome.

DNA Methylation Here we covered the main known gene regulation mechanisms, but there are more regulation methods which have been researched, and are note negligible. DNA methylation at cytosine and CpG sites has been involved in genome silencing in multiple processes (Jones et al. 2012), and has been documented as largely correlated with gene expression inhibition when present in promoters. In enhancer elements, anti-correlation was found between DNA methylation density and enrichment of active enhancer histone marks and TF binding (Stadler et al., 2011; Thurman et al., 2012), although the cause and consequence relationship underlying these correlations is not yet clear.

Epigenetics Limitation The currently most accurate method for predicting the location of tissue specific enhancers in a genome wide scale, is analyzing the histone marks and TF and cofactors presence using ChIP-seq from a cell line or from a tissue, combined with DNase I hypersensative (DHS).

Several approaches have faced the problem of locating enhancers by modeling gene expression based on epigenetic marks. However, these models rely on experimental data, and are inherently limited to the specific tissues we can extract and isolate for epigenetic examination. Furthermore, such models do no supply a classification of enhancers for new variation found in the population. Another disadvantage is the need for live cells for the verification of the regulatory activity of a sequence. The ultimate goal of an efficient computational method for predicting and explaining the reason for the functional nature of sequences "in-silico" has produced positive, yet far from sufficient results in the last years, as reviewed in (Kleftogiannis et al. 2016).

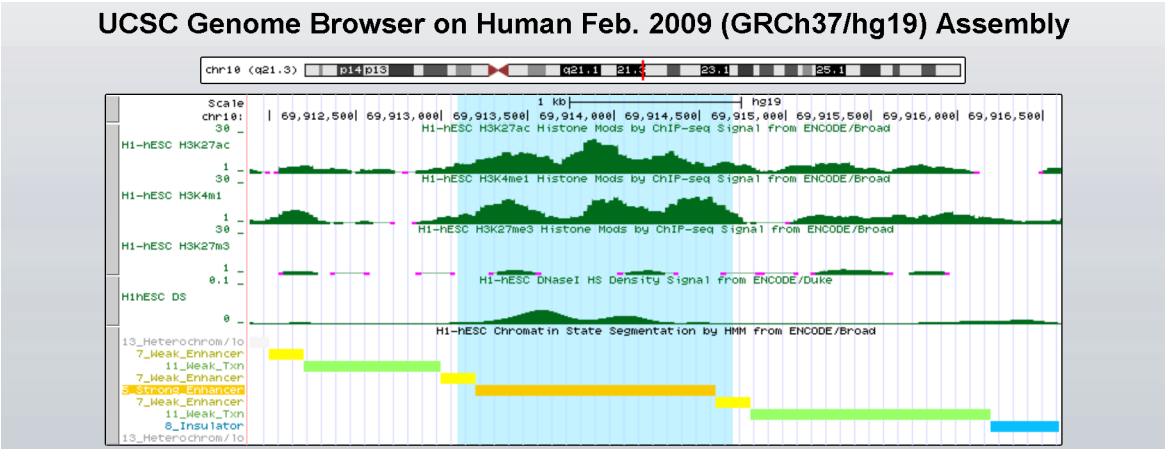


Figure 4: Epigenetic feature tracks measured by ENCODE, taken from the tenth chromosome of a H1-hESC cell line. Highlighted in light blue, the peaks of the H3K27ac (1st green plot) and H3K4me1 (2nd green plot) histone marks and the DNaseI hyper sensitivity feature (4th green plot) together with the lack of H3K27me3 (3rd green plot) signal are indication of an active enhancer, as indicated by the ChromHMM classification (bottom). Note the decrease between the two peaks of H3K27ac and H3K4me1 is located on top of the increase of the DNaseI hyper sensitivity, which implies a cleavage in between two nucleosomes with modifications.

Previous Work

There are several achievements in the task of predicting epigenetic and regulatory properties of DNA elements given only their sequence using machine learning algorithms. DeepSEA (Zhou and Troyanskaya, 2015) deep convolutional neural network (DCNN) is fed with 1000 bp DNA sequence and predicts an output vector of 919 binary features which represents the chromatin modifications of 200 bp bin in the center of the input sequence. The training labels used are the chromatin modification are extracted from ENCODE and Roadmap Epigenomics data releases.

Basset (Kelley et al. 2016) also used DCNN with known PWM as weights initialization on ENCODE and Roadmap Epigenomics data to predict a binary vector that represents accessibility in 164 cell types based on 600 bp DNA sequence. 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 kmers presence indicator vector were used as features for an SVM classifier to predict the role of DNA sequences with varying lengths.

ChromHMM (Ernst and Kellis, 2012) is a widely used software that tackles the problem of analyzing the epigenetic data for concluding roles in the genomic sequence. The algorithm uses chromatin mark reads, threshold to binary values, as input to HMM which then allows classifying the genome state in each position in the genome.

A disadvantage of these method is their need for a training data of known regulatory elements or with epigenetic data, which is commonly obtained from GWAS surveys done on 127 obtained human cell types in the Roadmap and ENCODE projects (Kundaje et al. 2015; Ernst et al. 2011). The number of different cell types in the human body is estimated to be higher than 2200 (Hatano et al. 2011, Diehl et al. 2016), where then number and location of tissue specific enhancers of the rest of the cell types is a mystery.

HOP-HMM extends the algorithm of (Kaplan et al. 2011) which also contains hidden states that emit TFBS sampled from PWMs to predict enhancers location in the genome. Both algorithms are part of the generalized hidden Markov model (gHMM) family, which are HMM variants that contain hidden states that may emit multiple observed variables. Although HMM variants using higher-order HMM, in which the transition and emission are dependent on previous hidden states where used previously (Ferguson, 1980; Preez, 1997), an HMM variant in which the emission is dependent on previous emissions is a less researched field.

Data Representation The DNA sequence, when read from cells, is usually stored in files, such as .fa, as a sequence of letters A,C,G and T. For an algorithm to process it, the characters are mapped into integers 1,2,3 and 4 respectively. For many algorithms, such as in DeepSEA, Basset, and our HOP-Baum-Welch, it is more suitable to represent the DNA sequence in a one-hot encoding as described in figure 3.

A common feature extraction technique is representing a DNA sequence as a vector of the in-sequence frequencies of all the possible kmer as used in gkm-SVM. In this technique, similarly to the bag of words technique in text analysis and natural language processing, the order of the kmer locations is sacrificed for a more meaning-oriented, structured and fixed-length data encoding.

Research Question

It has been shown in vivo (Visel et al. 2007) that the insertion of an enhancer sequence and an adjacent target gene will cause an activation of the target gene in mice. The transgenic mice shows the activation of the inserted enhancer although it was inserted to an arbitrary location in the genome and without epigenetic information. This implies that for the newly introduced sequences, their DNA sequence alone is potentially enough to make them active as enhancers. This strengthen the possibility that classifying sequences as enhancers could be achieved even without the epigenetic information, which is in missing for many types of cells.

Figure 5: Transgenic mouse with synthetic enhancer related dorsal root ganglion spinal neurons. The enhancer was inserted to the mouse genome with an adjacent blue color marker gene, which was activated even when no additional epigenetic data was introduced to the mouse. Taken from Vista Enhancer Browser.

My task of my research is to locate sequences with potential enhancer activity, in cell types we have not yet obtained epigenetic data from.

Markov Models

Markov Model Markov model (Markov, 1906) is a stochastic model named after Andrey Markov, a Russian mathematician. In a Markov model, at any time the model is at one of m states $\{S_1, ..., S_m\}$, where the first state is sampled from a distribution $\pi_i = P(y_1 = S_i)$ and the probability of transitions between the states is denoted by $T_{i,j} = P(y_t = S_i | y_{t-1} = S_j)$. The model's travel over the states is called a Markov process, and the sequence of 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 all following transition, which due to the independence between transition events can be written as :

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

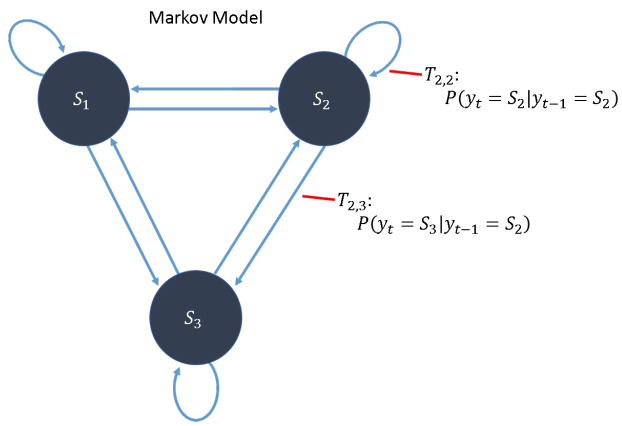


Figure 6: Markov model with 4 states, marked as S_1, S_2, S_3, S_4 with two of all possible transition probabilities written. T matrix describes the transition distributions, where $T_{i,*}$ is the distribution of the model's next step when in S_i.

Generative & Discriminative Models

There are a two main distinguished approaches in the machine learning models, the generative models and discriminative models. Both assumes an observed variables X and target variables Y, also commonly referred to as labels.

- The generative models assume a joint probability $P(X, Y)$. Using the data one can estimate the distribution $P(X, Y)$, then from it estimate $P(Y|X)$. It is assumed that such a model can generate the random instances of the data either as pairs of (x, y) or generate instances of x given y.
- Discriminative models assume conditional probability $P(Y|X)$, which is estimated directly from the data.

In classification problems, the task at hand is to arrive from the observed X to its label Y, e.g. given a DNA sequence X, deciding its role label Y. Both models eventually use the $P(Y|X)$ estimation to base their classification. Namely, classifying a data sample x by $y_{est} = argmax_y P(Y = y|X = x)$.

Discriminative models are more widely used than generative models. They are often easier to use and build since they require less assumptions on the origin or generation of the data. For example, a discriminative model such as a DNN classifying the role of DNA sequence assumes very little on the way the DNA sequence is related to it's role and generated based on it, but instead it finds features in the sequence that indicate its role. Such a model often gives very little for later understanding of the nature of the data generation process, and can generate no new data later for other uses.

Hidden Markov model

Multiple signal processing algorithms have been used in computational biology, and HMM is especially popular among them. Hidden Markov model (HMM) is a statistical model proposed by Leonard Baum (Baum et al. 1966) and is based on the Markov model for modeling regions with alternating frequencies of patterns and symbols. It was used in various engineering fields since the 1980s especially in speech recognition (Rabiner and Juang, 1993), character recognition and digital communication and was adopted in the computational biology field.

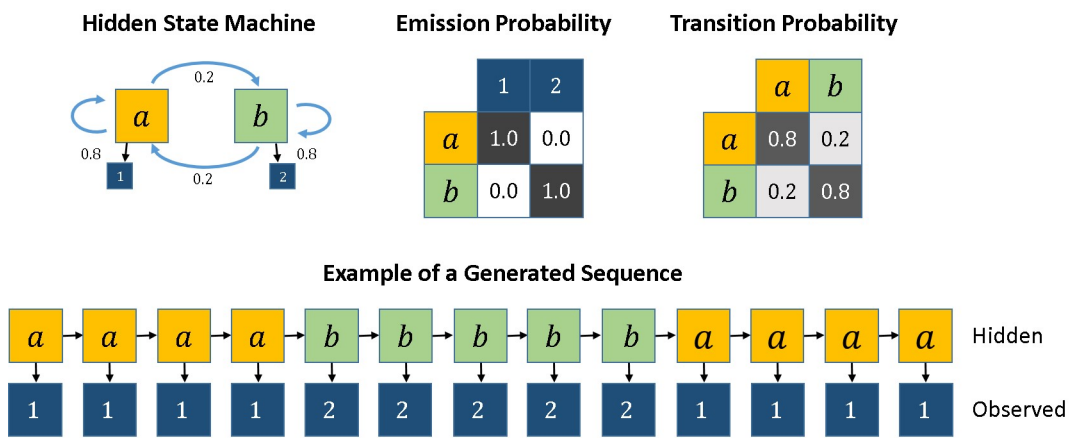


Figure 7: A toy example of HMM with 2 hidden states, the observed variables are emitted in a sequence, based on the hidden state at their location. The hidden variables sequences (orange and green on the bottom) is a two state Markov chain that emits observed variables on every step.

Hidden Markov model (HMM) is a model that travels over hidden states in a Markov process, and while doing so it emits variables called observed variables. As the Markov model, HMM is an generative model and it assumes the existence of a joint probability $P(x_{1:L}, y_{1:L})$ that is derived from the compact parameters θ. As a generative model, HMM relies on the assumption that the observed DNA sequence $X = x_1, ..., x_L$ can be generated by a parameterized model θ, and has an hidden state sequence $Y = y_1, ..., y_L$ that are generated alongside it. In this generation process, a single observed variable is emitted per step of the model, and so the observed sequence is generated with the same length as the hidden Markov chain. The observed variables $V_1, ..., V_n$ are sampled from an emission distribution $E_{i,j} = P(x_t = V_j | y_t = S_i)$, that is conditioned on the hidden state of the model. Similarly to the Markov model, the distribution to the first hidden state is marked asπ and the transition distribution is marked as T.

The likelihood of the observed sequence is important, as we explain later the maximum likelihood estimation algorithm that is used to fit the HMM parameters θ.

For example, assuming the DNA are composed of genes enhancers and background regions, with each having different nucleotide frequency, then we can say that the DNA sequence was generated by a HMM with underlying sequence of 4 hidden states: gene, promoter, enhancer and background where each has its own nucleotide frequency. The emitted observed DNA sequence X is determined by the underlying hidden sequence Y that describes the "mode" of the sequence in each position.

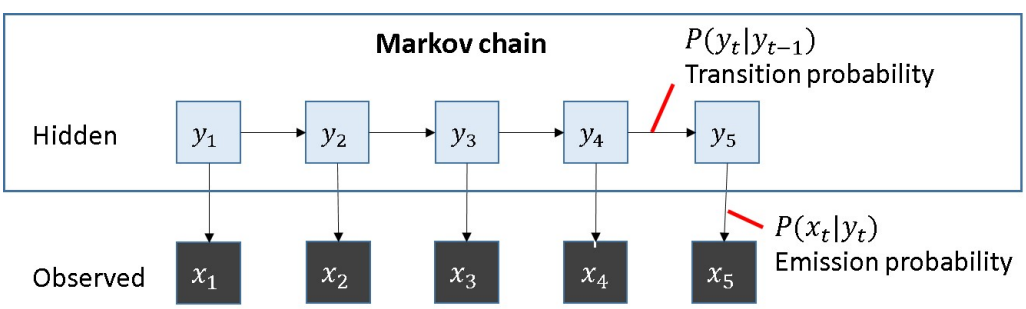


Figure 8: a HMM chain with length of 5. The underlying hidden states (up) are generated in a Markov process and each state emits an observed variable, sampled from the state's emission probability.

<u>HMM Limitations</u> [preposition, motivation, regularization]

Although HMM is simple and efficient, applying it on DNA sequences has a major disadvantage which is the inherit Markovian lack-of-memory property. This property means that on every step of the model, the next state is dependent only on the previous state, without further history consideration. For the task of emitting a motif, where

each position has a different emission distribution depending on the location in the motif, a HMM model would need to contain 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 the fixed-sized genome data.

A common way to avoid overfitting the data when training machine learning models is regularization the model's parameters, a method we can incorporate while training HMM as well. A possible regularization is constraint on the learned transition matrix T to be sparse. Similarly, the 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. 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.

There is a trade-off regarding the use of high-order emission compared to high-order transition, or in other words whether the transition or the emission probability is conditional to the previous states and emissions. Both can emit sequences with structure of higher order, for example both can describe a sequence where C will never come after T, as well as sequences containing kmers, etc. The high-order transition allows more freedom and specificity in the shape of a per-location probability of a transition to another, unrelated state depending on previous states, with the cost of more parameters. This transition seemed unnecessary since we assumed the transition to another state, or enhancer in our case, is only loosely conditional to the last several emissions.

HOP-HMM

Here we present HOP-HMM, a variant model of HMM, that is well fitted to utilize the structure of enhancers containing TFBSs inside them, due to the TFBS emitting sub-states that take part in the generation process of the sequence. The base feature of HOP-HMM is its two types of hidden states: PWM sub-state, and base-states a base-state that can transfer into one of it's PWM sub-states.

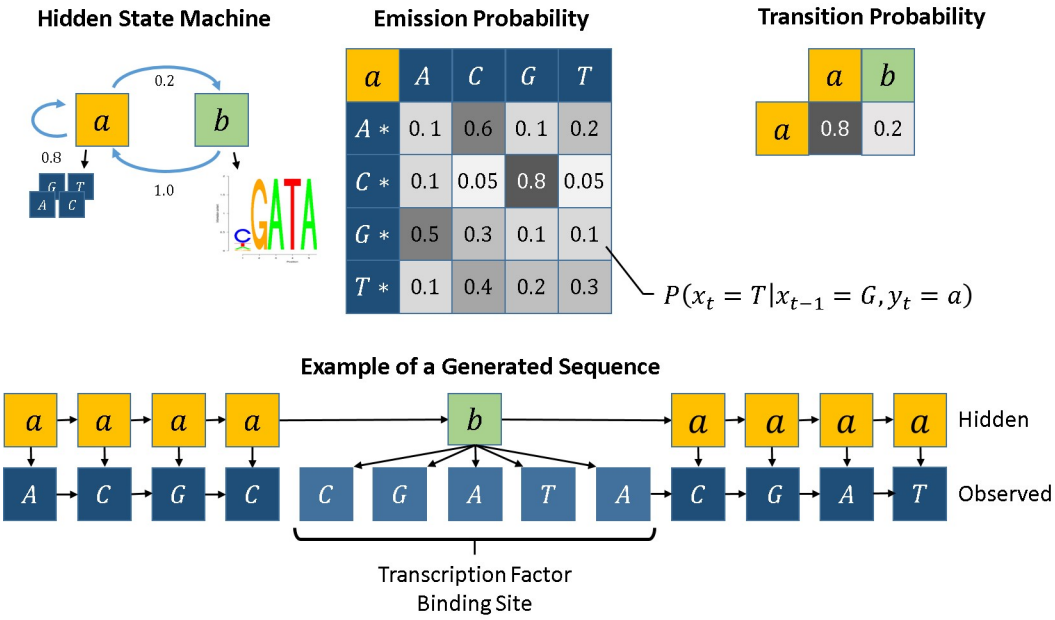


Figure 9: HOP-HMM toy example with one base-state 'a', which emits a single observable DNA variable at a time, and one sub-state 'b' which emits a TFBS sampled from its PWM. The emission of the base state 'a' is of order one (o=1), therefore it is conditioned on the last previous observable variable.

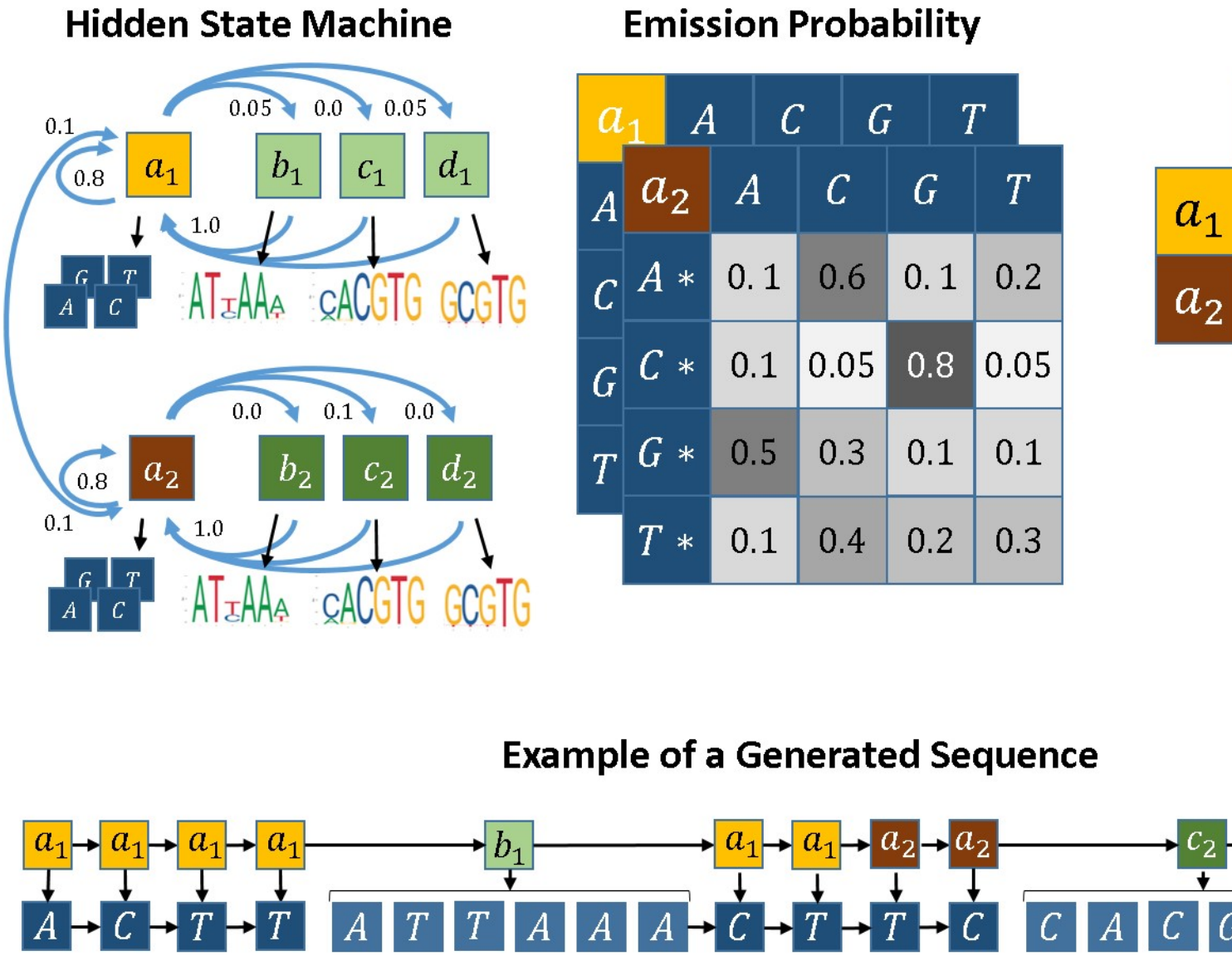


Figure 10: HOP-HMM with two base-state a_1 and a_2, where each has 3 TFBS emitting sub-states. The example generated sequence is built out of two types of sequences, each has its own TFBS frequency and background nucleotide bigram frequency, representing two alternating types of enhancers.

Hidden States

The HOP-HMM generates sequences with m different base-state, where each one has its own k sub-states. For example, in figure 10 we see a HOP-HMM with $m = 2$ and $k = 3$. While most base-states represent an enhancer type, we also would like to have background regions in between the enhancer. For that end we may predefine one or more base-states as background, by restricting the probability of transferring from these base-states into their sub-states.

We use two indices to describe a hidden state:

- base-states are indexed as $(j, 0)$ where $j \in [m]$
- sub-states are indexed as (j, l) where $j \in [m]$ and $l \in [k]$.

The (j, l) sub-state belongs to the $(j, 0)$ base-state (see figure 11), i.e. it can be transferred into by the model only from $(j, 0)$. Note that we used simpler notation in figures 9 and 10 for readability.

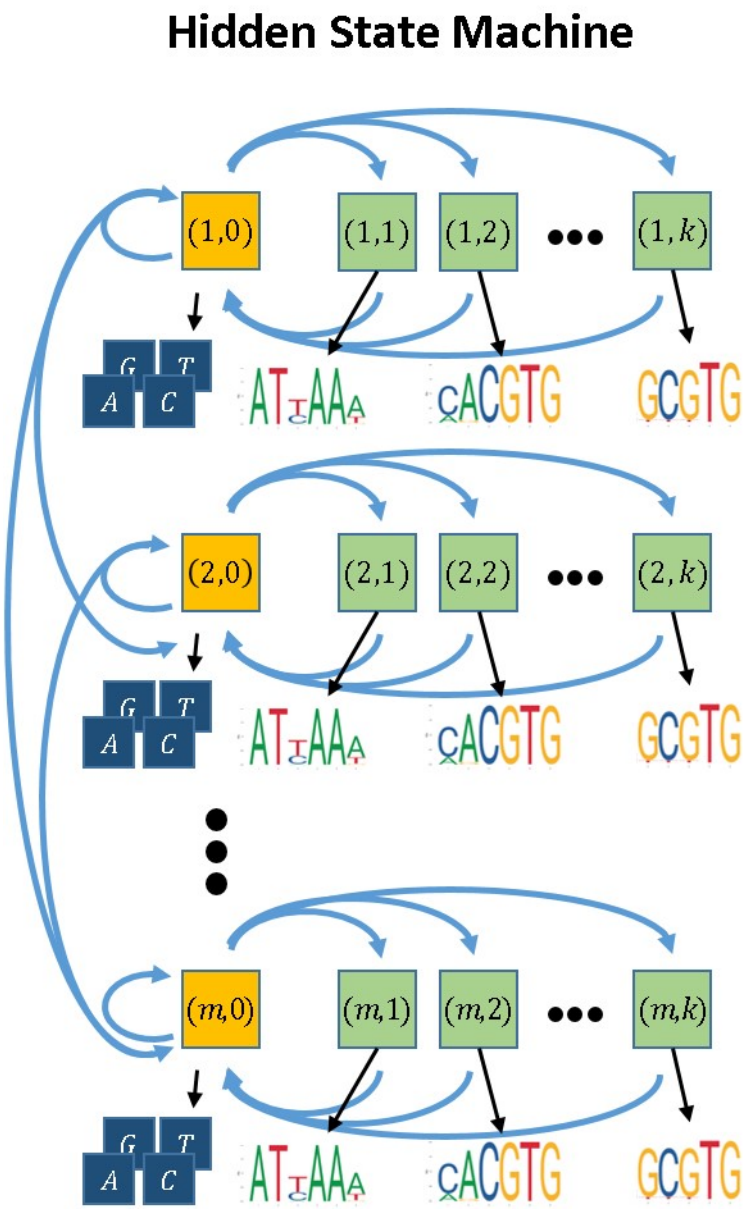
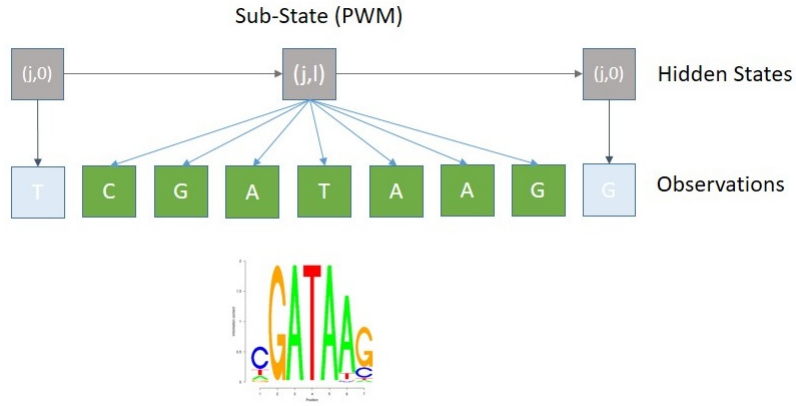


Figure 11: general hidden states graph of HOP-HMM. Each row represents a sequence type, where each of the m base-states (yellow) has k sub-states (green). Not all transitions are possible, moving between the rows is possible only by a base-state to base-state transition.

Emission

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 sub-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 sub-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 sub-state, it emits a motif by sampling from a PWM column by column independently, as described in Figure 3.



[tommy: remove the indices?]

Figure 3: PWM emission of sub-states. PWMs are a set of per-position nucleotide emission probabilities, that are sampled independently to create a TFBS motif.

The base-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, base-states emit single nucleotides, where their emission is conditional on the previous of letters emitted in the DNA sequence. The emission from base-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 $E_{j,x_{t-o+1},x_{t-o+2},...,x_t} = P\left(x_t | y_t = (j,0), x_{t-o+1},...,x_{t-1}\right)$, 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 $\sum_{i\in[4]} \frac{E_{j,i,x_1,...,x_{t-1}}}{4}$.

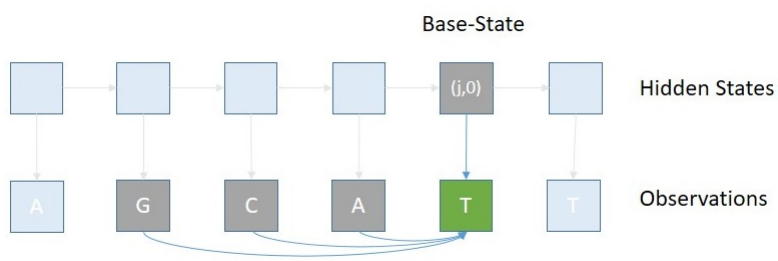


Figure 2: high-order emission of base-states. Each emission is dependent on the hidden base-state and o-1 previous observations.

Transition

In HOP-HMM, the first hidden state in a sequence can only be a base-state. The first base-state, as in HMM, is chosen by sampling from π, a probability vector $\pi_j = P(y_1 = (j,0))$. Once in a base-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 4, we hold only the possible transition probabilities in two matrices, representing the two types of allowed transitions:

T for base-state to base-state transitions, a $m \times m$ matrix where $T_{j_1,j_2} = P\left(y_{t+1} = (j_2,0) | y_t = (j_1,0)\right)$.

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

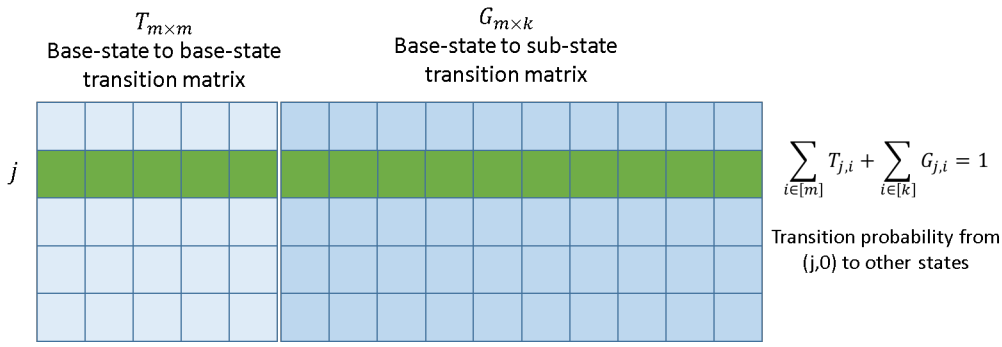


Figure 4: a compact way to hold the allowed transition probabilities. Concatenation of a row in T and G holds the probability of the next state given the current one.

HOP-HMM Denotations

For clarity, the denotations used for the HOP-HMM algorithm:

o - order of base-state emission, emit distribution of variable depends on $o-1$ last variables (hyper parameter)

m - number of base-states (hyper parameter)

k - number of sub-states for each base-state (hyper parameter)

L - length of the DNA sequence (data given to the algorithm)

W_i - the PWM of the i'th sub-state of all base-states in the model (hyper parameter)

Baum-Welch Algorithm

HMM Likelihood Function

In the maximal-likelihood estimation problem before us, we have the observed $x_{1:L}$ and we would ultimately like find the parameters that maximize the likelihood of it:

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

This likelihood function $L\left(\theta|x_{1:L}\right)$, also called the incomplete-data likelihood function, could be written, using the total probability law, as:

$$L\left(\theta|x_{1:L}\right) = P_\theta(x_{1:L}) = \sum_{y\in[m]^N} P_\theta\left(x_{1:L}, y_{1:L}\right) \tag{1}$$

The probability on the right side $P_\theta(x_{1:L}, y_{1:L}) = L(\theta|x_{1:L}, y_{1:L})$ is called the complete-data likelihood function, and in the case of the basic HMM it is calculated by:

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

In our variation HOP-HMM we assume different transitions and emissions hence the complete-data likelihood requires different calculation while the rest of the EM algorithm still holds.

Unfortunately, optimizing or calculating the incomplete-data likelihood in (1) involves a summation of exponential-by-L elements (L is the length of the DNA sequence), which is infeasible. Instead, the strategy of the EM algorithm is to optimize the expected value of the complete-data log-likelihood $log\left(P\left(x_{1:L}, y_{1:L}|\theta^{'}\right)\right)$ 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, as it is our DNA sequence. For this task we can formally define our target function Q:

$$Q\left(\theta, \theta^{'}\right) = E_Y\left[log\left(P_\theta\left(x_{1:L}, y_{1:L}\right)\right)|x_{1:L}, \theta^{'}\right] = \sum_{y\in[m]^N} log\left(P_\theta\left(x_{1:L}, y_{1:L}\right)\right) P_{\theta'}\left(x_{1:L}, y_{1:L}\right) \tag{3}$$

Note that here, E is expressing an expected value and not the HMM emission probability. Every EM iteration is built of two parts: the E-step and the M-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. Although this seemingly still requires an exponential summation, we can use a dynamic programming approach to overpass it, with the cost of $O(N \cdot m)$ memory usage.

Using equations (2) and (3) allows us to split the Q function to three independent parts.

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

then by manipulating the summation and the state sequence cases 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 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)

The first part is maximized using Lagrange multiplierλ:

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

we derive the equations and get $\frac{P_{\theta'}(x_{1:L}, y_1=j)}{\pi_j} = \lambda$ for $j \in [m]$. Then we multiply each equation by the denominator and sum these m equations to receive $\lambda = P_{\theta'}\left(x_{1:L}\right)$, which yields:

Algorithm 1 Forward Algorithm

Input:

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

Algorithm:

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

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

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

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

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

$$\pi_j = \frac{P_{\theta'}\left(x_{1:L}, y_1 = j\right)}{P_{\theta'}\left(x_{1:L}\right)} = P_{\theta'}\left(y_1 = j | x_{1:L}\right) \tag{4}$$

We may follow the Lagrange multipliers method in the second and third parts, from which we'll receive:

$$T_{j_1,j_2} = \frac{\sum_{t \in 2...L} P_{\theta'}\left(x_{1:L}, y_{t-1} = j_1, y_t = j_2\right)}{\sum_{t \in 2...L} P_{\theta'}\left(x_{1:L}, y_{t-1} = j_1\right)} = \frac{\sum_{t \in 2...L} P_{\theta'}\left(y_{t-1} = j_1, y_t = j_2 | x_{1:L}\right)}{\sum_{t \in 2...L} P_{\theta'}\left(y_{t-1} = j_1 | x_{1:L}\right)} \tag{5}$$

and

$$E_{j,b} = \frac{\sum_{t \in [L]} P_{\theta'}\left(x_{1:L}, y_t = j\right) \mathbf{1}_b(x_t)}{\sum_{t \in [L]} P_{\theta'}\left(x_{1:L}, y_t = j\right)} = \frac{\sum_{t \in [L]} P_{\theta'}\left(y_t = j | x_{1:L}\right) \mathbf{1}_b(x_t)}{\sum_{t \in [L]} P_{\theta'}\left(y_t = j | x_{1:L}\right)} \tag{6}$$

where $\mathbf{1}_b(x_t) = \begin{cases} 1 & b = x_t \\ 0 & otherwise \end{cases}$

After stating the intentions of each EM iteration in (4), (5) and (6), we now need to calculated them in order to successfully learn the HMM parameters. Specifically, notice that to resolve all the parameters update states in (4), (5) and (6), it is enough to calculate the two terms during the E-step:

$$P_{\theta'}\left(y_t = j | x_{1:L}\right) \tag{7}$$

$$P_{\theta'}\left(y_{t-1} = j_1, y_t = j_2 | x_{1:L}\right) \tag{8}$$

We will calculate these terms by using the Forward-Backward algorithm. The Forward-Backward algorithm (Rabiner, 1989) is a method to dynamically calculate two matrices, α and β, both of size $m \times L$.

Forward Algorithm

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

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

It is calculated the dynamic algorithm:

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 its hidden state that emitted it y_t.

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

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

Algorithm 2 Backward Algorithm

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

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

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

Backwards 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:

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

It is calculated by the dynamic algorithm:

This matrix building process is similarly explained by:

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

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

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

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

Auxiliary Probabilities

Having the forward and backward probability matrices α and β, we now have all that is needed to calculate (8) and (9), using once more the HMM conditional independence in our calculations. Here we used the fact that

We denote the terms values of (7) as γ, a matrix of size $L \times m$:

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

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

And we denote the values of (8) as ξ, a matrix of size $L - 1 \times m \times m$:

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

$$= \frac{P\left(y_{t-1} = j_1, x_{1:t-1}\right) \cdot P\left(y_t = j_2 | y_{t-1} = j_1\right) \cdot P\left(x_t | y_t = j_2\right) \cdot P\left(x_{t+1:L} | y_t = j_2\right)}{\sum_{j' \in [m]} P\left(y_t = j, x_{1:t}\right) \cdot P\left(x_{t+1:L} | y_t = j\right)} =$$

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

Baum-Welch Algorithm for HOP-HMM

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 calculate and optimize a Q function:

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

where $L_W(\overline{x})$ is the likelihood of the TFBS $\overline{x}$ to be emitted by PWM W: $L_M(\overline{x}) = P(\overline{x}|W) = \prod_{i \in \{1,\ldots,|\overline{x}|\}} W_{\overline{x}_i,i}$. The last component, with the TFBS likelihood does not contain elements from θ, therefore it is not optimized in the m-steps.

The θ which optimizes Q, i.e. $\theta_{max} = argmax_\theta Q(\theta, \theta')$, is built similarly as in the regular Baum-Welch algorithm, as following:

$$\pi_j = \frac{P_{\theta'}\left(x_{1:L}, y_1 = (j, 0) | \theta^{'}\right)}{P_{\theta'}\left(x_{1:L} | \theta^{'}\right)} = P_{\theta'}\left(y_1 = (j, 0) | x_{1:L}\right) \tag{9}$$

$$T_{j_1,j_2} = \frac{\sum_{t \in 2 \ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j_1, 0), y_t = (j_2, 0)\right)}{\sum_{t \in 2 \ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j_1, 0)\right)} = \frac{\sum_{t \in 2 \ldots L} P_{\theta'}\left(y_{t-1} = (j_1, 0), y_t = (j_2, 0) | x_{1:L}\right)}{\sum_{t \in 2 \ldots L} P_{\theta'}\left(y_{t-1} = (j_1, 0) | x_{1:L}\right)} \tag{10}$$

$$G_{j,l} = \frac{\sum_{t \in 2 \ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j, 0), y_t = (j, l)\right)}{\sum_{t \in 2 \ldots L} P_{\theta'}\left(x_{1:L}, y_{t-1} = (j, 0)\right)} = \frac{\sum_{t \in 2 \ldots L} P_{\theta'}\left(y_{t-1} = (j, 0), y_t = (j, l) | x_{1:L}\right)}{\sum_{t \in 2 \ldots L} P_{\theta'}\left(y_{t-1} = (j, 0) | x_{1:L}\right)} \tag{11}$$

and

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

Algorithm 3 HOP Forward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

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

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

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

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

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

$$= \frac{\sum_{t \in o,...,L} P_{\theta'}\left(y_t = (j,0)|X\right) \mathbf{1}_{b_1,...,b_o}(x_{t-o+1,...,t})}{\sum_{t \in o,...,L} P_{\theta'}\left(y_t = (j,0)|x_{1:L}\right)}$$

Hence to complete the EM iteration, the three missing components in (9),(10),(11),(12) are:

$$P_{\theta'}\left(y_t = (j,0)|x_{1:L}\right) \tag{13}$$

$$P_{\theta'}\left(y_{t-1} = (j_1,0), y_t = (j_2,0)|x_{1:L}\right) \tag{14}$$

$$P_{\theta'}\left(y_{t-1} = (j,0), y_t = (j,l)|x_{1:L}\right) \tag{15}$$

For the calculation of these probabilities, we will need to calculate the forward and backward probabilities, and then few other auxiliary terms. The forward and backward probabilities are only for indicating the sequence entering-to and exiting-from base-states:

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

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

HOP-Forward Algorithm

We calculate α of size $m \times L$, iterating over $t = 1, 2, ..., L$ as following:

Calculation notes: 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 $E_{j,x_1,...,x_t} = \sum_{b_1,...,b_{o-t} \in \{A,C,G,T\}} \frac{1}{4^{o-t}} \cdot E_{j,b_1,..,.b_{o-t},x_1,...,x_t}$ that is the expected probability upon possible preceding variables. 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, $P(x_i) = \frac{1}{4}$ where $i < 1$. Also, when summing the sub-state transition part $l \in [k]$, the PWMs W_l with length equal or bigger than $t+1$ are not part of the summation.

Algorithm 4 HOP Backward Algorithm

Input:

X - Observed DNA sequence

Algorithm:

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

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

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

$$\beta_{j,t} = \underbrace{\sum_{j' \in [m]} \beta_{j',t+1} \cdot E_{j',x_{t-o+2},...,x_{t+1}} \cdot T_{j,j'}}_{\text{base-state transitions}}$$

$$\underbrace{+\sum_{l \in [k]} \beta_{j,t+|W_l|+1} \cdot L_{W_l}\left(x_{t+1}, ..., x_{t+|W_l|}\right) \cdot E_{j,x_{t-o+|W_l|+2},...,x_{t+|W_l|+1}} \cdot G_{j,l}}_{\text{sub-state transitions}}$$

HOP-Backward Algorithm

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

Note that when $t > L - |W_l|$, there are missing observable variables to fully calculate the sub-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 PWM sub-state when the PWM is too long to fit into the sequence X length.

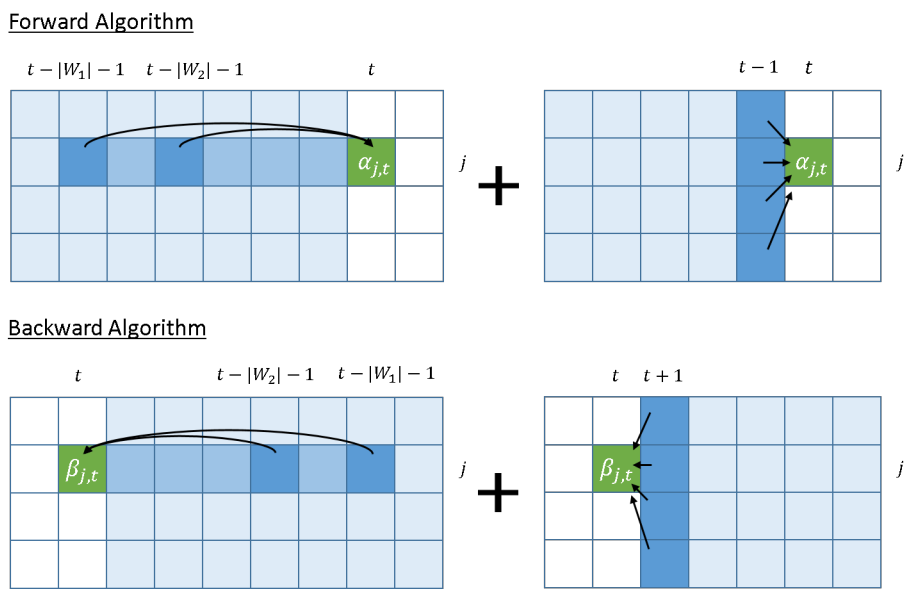


Figure 5: In our version of the forward-backward algorithm, each of the α and β cells are filled from both the adjacent base-states transitions and the base-states preceding or proceeding the motifs emitted by the sub-states.

Forward Algorithm Explanation

We will now explain why the described dynamic calculation result with $\alpha_{j,t} = P\left(y_t = (j,0), x_{1:t}\right)$ and $\beta_{j,t} = P\left(x_{t+1:L} | y_t = (j,0)\right)$, 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 base-state (j,0):

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

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

We could develop the right term of a sub-state transition using the chain rule:

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

Because of x_t is dependent on only y_t (and also $x_{t-o:t-1}$ if y_t is a base-state) and y_t is dependent on only y_{t-1}, we can simplify the probabilities:

$$\begin{aligned} P\left(y_{t-|W_l|:t-1} = (j', l), y_{t-|W_l|-1} = (j, 0), x_{1:t}\right) = & P\left(y_{t-|W_l|-1} = (j, 0), x_{1:t-|W_l|-1}\right) \\ & \cdot P\left(y_{t-|W_l|:t-1} = (j, l) | y_{t-|W_l|-1} = (j, 0)\right) \\ & \cdot P\left(x_{t-|W_l|:t-1} | y_{t-|W_l|:t-1} = (j, l)\right) \\ & \cdot P\left(x_t | y_t = (j, 0), x_{t-o:t-1}\right) \end{aligned}$$

We can now replace the received terms with the components ofθ and with already filledα cells:

$$P\left(y^i_{t-|W_l|:t-1} = (j, l), y^i_{t-|W_l|-1} = (j, 0), x^i_{1:t}\right) = \alpha_{j,t-|W_l|-1} \cdot G_{j,l} \cdot L_{W_l}\left(x_{t-|W_l|}, ..., x_{t-1}\right) \cdot E_{j,x_{t-o+1},\ldots,x_t}$$

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

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

Using the conditional independencies to simplify the probabilities:

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

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

Backward Algorithm Explanation

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 base-state emission is conditional on them.

Using the law of total probability:

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

$$= \underbrace{\sum_{j'} P\left(y_{t+1} = (j',0), x_{t+1:L} | y_t = (j,0), x_{t-o+2:t}\right)}_{\text{base-state transition}} + \underbrace{\sum_{l} P\left(y_{t+1:t+|W_l|} = (j,l), y_{t+|W_l|+1} = (j,0), x_{t+1:L} | y_t = (j,0)\right)}_{\text{sub-state transition}}$$

For the base-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 sub-state transition term, we use once more the chain rule, followed the simplification using the conditional independencies:

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

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

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

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

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

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

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

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

Auxiliary Probabilities

Using the forward and the backward probability matrices, we are ready to calculate the auxiliary probabilities (13) (14) (15). The first probability that will help us for that task is ψ, a matrix of size $m \times k \times L$:

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

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

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

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

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

The second probability, is of the observed sequence X:

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

Now we can calculate probability (13) of the base-state at a given position given the sequence X, denoted as γ of size $m \times L$:

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

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

The probability (14) is the base-state to base-state transition given the sequence X, denoted as ξ of size $m \times m \times L$:

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

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

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

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

Finally, the probability (15) is the base-state to base-state transition given the sequence X, denoted as ξ of size $m \times k \times L$:

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

Now with (13), (14), (15) at hand, we can calculate θ_{max} by assigning them at (9), (10), (11), (12).

Algorithm 5 HOP Baum-Welch

Input:

X - Observed DNA sequence

Algorithm:

for s=1...MAX_EM_ITERATIONS:

//e-step

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

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

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

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

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

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

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

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

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

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

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

If θ converged, break EM for loop

HOP Baum-Welch

To conclude, the total EM algorithm for HOP-HMM:

Learning Multiple Sequences at Once

The algorithm here is described for learning a single sequence of observable variables X. For learning the parameters θ from multiple sequences at once, we can use the same method as introduced in the original paper of Baum-Welch algorithm (Rabiner, 1989), which calculates the E step probabilities for every sequence, and in the m-step sums all positions from all sequence for the parameters update.

Classifying DNA a Sequence

One of the goals of the HMM and HOP-HMM learning process described above is to be able to classify the DNA sequences by deciding their hidden-state per location. After the Baum-Welch algorithm procedure, we might want to reach that goal by choosing per position t the max-likelihood hidden-state. In the HOP-HMM case, this means getting the hidden sequence:

$$y_t^* = argmax_{(j,l)} P\left(y_t = (j,l) | x_{1:L}\right)$$

Algorithm 6 HOP-Viterbi Algorithm

Input:

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

X - Observed DNA sequence

Algorithm:

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

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

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

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

    create two vectors:

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

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

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

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

$y_L = \left(argmax_j V^1_{j,L}, 0\right)$

t=L

while t > 1:

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

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

    $y_{t-1} = (j, 0)$

    $t = t - 1$

  else:

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

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

    $t = t - |W_l| - 1$

Although simple to calculate with the γ auxiliary probability, such a hidden sequence could be problematic. Each hidden-state maximizes the likelihood at its location, but the hidden sequence together with its transitions might not be the maximal likelihood one, and in fact might even include illegal transition (with probability 0).

Our task here is reaching the hidden sequence that maximizes the likelihood of the observed sequence, where all the sequence is considered:

$$y^*_{1:L} = argmax_{y_{1:L}} P\left(y_{1:L} | x_{1:L}\right) \tag{16}$$

HOP-Viterbi Algorithm

A common way to derive the hidden sequence that maximizes (16) is the Viterbi algorithm, which is similar to the Forward algorithm. The main differences between the two algorithms are the usage of *max* instead of summing over the possible transitions on each step of the dynamic algorithm, and keeping information of the chosen maximal value used via the *argmax* function. The HOP-HMM adaptation to this algorithm includes supporting the PWM sub-states and the high order emission of the base states.

Results

Synthetic Sequences

We use data synthesized by a HOP-HMM, to show that the parameters learned in the HOP Baum Welch algorithm are similar to the ones that were used in the generation. For each test we generated 1000 sequences of length 10,000 using a model with θ parameters, synthetically generated. The models contained different number of base-states and sub-states in each run, and some of the base-states where set to behave as "background base-states". Background

regions typically have lower occurrences of TFBS and are longer, compared to enhancers. For simulating such a behavior, the background base-states have a low probability to enter into their PWM sub-states and have higher probability to move into them and staying in them compared to non-background base-state.

Pretraining When sequences are labeled as tissue-specific enhancers are available, it is possible to pretrain a multi-base-state HOP-HMM with them. The E and G parts of such a model could be initialized with parameters taken from a smaller HOP-HMM, trained on 2 class datasets built out of tissue-specific enhancers and background DNA sequences. For each tissue, we build a dataset and trained a 2 base-state HOP-HMM model (one base-state for the enhancer and one for the background) and used the learned parameters of the enhancer base-state to initialize the bigger multi-base-state HOP-HMM.

[TODO: generate such a figure with 12 scatter plots. change those constants in each plot so an impact is shown. consider removing the pretraining if doesn't improve (and mention it was tried and it didn't work)]

[TODO: more scatter plots with more \ less PWMs in enhancers, more \ less PWMs joint between enhancer types, different base-states emission order, with \ without E sharing.] ***Figure: Comparison of learned and real parameters values of 3 different HOP-HMMs. The comparisons are done after learning 10%, 50%, 100% of generated sequence. The right most plots show a comparison after learning 100% of generated sequence using a pretrained initialization of the parameters. (a) 3 base-states, and 5 sub-states for each base-state. (b) 7 base-states out of which 1 is a background base-state, and 10 sub-states for each base-state. (c) 10 base-states out of which 2 are background base-states, and 30 sub-states for each base-state.***

[TODO: generate sequence figure] ***Figure: Hidden states generated synthetically by a HOP-HMM (top), hidden states estimated by the HOP-Viterbi algorithm (middle) and base-state posterior probability $\gamma_{j,t} = P(y_t = (j, 0)|X)$ (bottom). The hidden states estimation and base-states probability are calculated using the parameters θ learned by the HOP-Baum-Welch algorithm, using the observable synthetic DNA sequence as input.***

[TODO: add to the figure caption the model's hyper-parameters m, k, etc .]

Roadmap Dataset

Perprocessing

Choosing the PWMs Part of the definition of a HOP-HMM is a set of PWMs emitted by the sub-states. JASPAR has 719 vertebrates PWMs, out of which we would want to use the most indicative for the tissue specific enhancers. The ideal PWM would be a long motif, strongly distinguishable from the background nucleotide probability and that occurs multiple times in the enhancer, and only in the enhancers that are active in a single tissue.

##########################

[tommy: position of the motifs]

[tommy: aligned heatmap sequences with the peak of the H3K27ac and post. prob., are they similar?]

TODO: comparison to ChromHMM and roadmap classification

TODO: Whole genome classification

TODO: comparison of HOP-HMM to SOTA

Misc

TODO: abstract

The TF inside the nucleus of specific tissues are thought to be a key factor in the activation of specific enhancer. The TFs form a transcription complex and are connected to the enhancer and promoter sequences on top of the TF binding sites (TFBS). Studies using TFBS of TF present in specific cell types are used to classify cell specific enhancer sequences. show heat map of AUC-ROC results. Between these TFBS, k-mer frequencies varies between enhancers and non regulatory "background" DNA, and was used to classify enhancers from background using only the k-mer distribution (Inbar and tommy, gkm-SVM), and is thought to play a role in spacial properties, nucleosome location and cleavage that cause accessibility of near-by TFBS. Using 44 out of 127 epigenetic data of Roadmap Project to select tissue specific enhancer sequences dataset. In our method, we look for different TFBS and k-mer presence in sequences to classify cell-specific sequences, inside regulatory modules.

TODO: Semi-Supervised Learning Scheme:

1. **Pre-training:** Calculate maximal likelihood initialization parameters θ_0 with from observed labeled dataset of sequences Y_0 and X_0
2. **Unsupervised learning:** Learn the θ given X unlabeled sequences by approximating $\theta_{best} = argmax_\theta \mathcal{L}(\theta|X)$
3. **Predicting labels:** Given learned θ, infer hidden states Y for unlabeled X
4. **Tuning:** Preform hyperparameters optimization

experiment graphs:

per nucleotide binary classification (background vs enhancer) - heatmap (x-location relative to peak, y-sequence index, color-post. probability) use the $\gamma_{i,*}$ of enhancers where i is the state of the enhancer, pick only the 2000+- around the enhancer's peak, where the center is the max of the H3K27ac signal, showing the sequences are most likely enhancers surrounded by non enhancers .

per sequence mutli-class classification (background vs enhancer) - heatmap (state number, y-sequence index, color-maximal post. probability for the sequence of that state) of γ of enhancers, where the center is the max of the H3K27ac, showing the sequences are most likely enhancers surrounded by non enhancers

per sequence classification

[per sequence binary classification - background vs enhancer]

[per nucleotide binary classification - background vs enhancer]

[per sequence binary classification - background vs enhancer]

TODO: Possible Applications

labeled enhancer seqs from multiple motifs-> EM to learn E M F per floor + setting$T = \mathbb{I}_{m\times m}$ -> posterior of whole genome with sliding window -> classify whole genome

learn E M F -> check correlation with TF expression

run EM on whole genome -> posterior of whole genome -> check correlation of posterior to ChIP-Seq of histone modifications

E M F T-> posterior of whole genome -> see if known critical SNPs are critical in classification