Comparison of Transcription Factor Binding Site Models

Thesis by

Md. Shariful Islam Bhuyan

Submitted in Partial Fulfillment of the Requirements for the degree of

Masters of Science

King Abdullah University of Science and Technology

Division of Mathematical and Computer Science and Engineering

Computer Science Program

Thuwal, Makkah Province, Kingdom of Saudi Arabia

May, 2012

The thesis of Md. Shariful Islam Bhuyan is approved by the examination committee.

Committee Chairperson: Professor Vladimir Bajic

Committee Member: Dr. Xin Gao

Committee Member: Dr. Xiangliang Zhang

ABSTRACT

Comparison of Transcription Factor Binding Site Models

Md. Shariful Islam Bhuyan

Modeling of transcription factor binding sites (TFBSs) and TFBS prediction on genomic sequences are important steps to elucidate transcription regulatory mechanism. Dependency of transcription regulation on a great number of factors such as chemical specificity, molecular structure, genomic and epigenetic characteristics, long distance interaction, makes this a challenging problem. Different experimental procedures generate evidence that DNA-binding domains of transcription factors show considerable DNA sequence specificity. Probabilistic modeling of TFBSs has been moderately successful in identifying patterns from a family of sequences. In this study, we compare performances of different probabilistic models and try to estimate their efficacy over experimental TFBSs data. We build a pipeline to calculate sensitivity and specificity from aligned TFBS sequences for several probabilistic models, such as Markov chains, hidden Markov models, Bayesian networks. Our work, containing relevant statistics and evaluation for the models, can help researchers to choose the most appropriate model for the problem at hand.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Professor Vladimir Bajic for suggesting this topic and for his invaluable feedback, encouragement and guidance throughout this work. His enthusiasm for research made my study very enjoyable and exciting and ultimately fruitful with rich experience. I would also like to thank him for enabling me to work in an amazing research atmosphere.

I also would like to extend my thanks to Wail Ba-Alawi for discussions and great co-operations in this work.

I would also like to express my deep gratitude to my parents and my wife for their continuous encouragement during this journey and their deep moral support at all times.

If it was not for the generous gift of the King Abdullah of Saudi Arabia, this work would not happen. I am truly grateful for the opportunity given to me.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Chapter I

INTRODUCTION

"Biology is so digital, and incredibly complicated, but incredibly useful. Biology easily has 500 years of exciting problems to work on, its at that level." – Donald E. Knuth.

Great discoveries at the molecular and genomic level opened new fields of research where computing is absolutely essential. Current assumption is that almost all of the biological functionality and processes of our life depends on the information stored inside the cell, the fundamental building block of any organism. This information is encoded as a macromolecule named Deoxyribonucleic acid (DNA). DNA is a long polymer of another molecule called nucleotides. Depending on the characteristics, we can categorize DNA building nucleotides into four types, Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). Computationally we can think of DNA as a string of billions of characters where the alphabet is {A,C,G,T} [15].

Not all the regions of DNA encodes information for protein synthesis, the major operational unit at molecular level inside cell, or Ribonucleic acids (RNAs), the molecules with diverse functionalities. The DNA regions with this important information are called genes. Different genes encode for different proteins or for non-coding RNAs. Moreover, not all the time this information is used. When cells produce a protein using the information from a corresponding gene, we say that the gene has expressed. A specific gene will express when

some special regulatory proteins called transcription factors (TF) [12] bind to TFBSs within regulatory regions of the genome. TFBSs serve as the intermediate framework for gene transcription regulation and a biological signal for TFs. Normally TFBS is a short sequence of nucleotides (typically 8-15). Several genes can have near similar TFBSs. So a single TF can participate in transcription process of several genes. On the other hand, different TFs are required to bind to different TFBSs in combination to initiate the expression of a single gene. TF bind to selected DNA segments (TFBSs) based on the physical affinity. So, genes co-transcribed due to the activity of the same TF can have a class of nearly similar TFBS patterns commonly called motif. A possible explanation is that a specific class of motifs has a single common ancestor from which the current motifs are derived by small mutation at the positions of non-essential nucleotides. When we discover the DNA sequence of a new organism, we need to identify its genes and associated regulatory motifs. There are several experiments to determine TFBSs and corresponding TFs and both commercial and public databases are available and contain such information e.g. TRANSFAC, Jasper etc.

Central dogma of molecular biology [10] describes how information flows at molecular level in cell to facilitate cellular activities e.g. signaling, cell cycle, metabolism. According to the Dogma, DNA is the container of instruction inside cell. A process called transcription reads a specific region of DNA when necessary and produces a complementary RNA. This one is processed to form the messenger RNA (mRNA) which then moves to Ribosome, a cellular organelle that translates mRNA to protein, a polymer of amino acids, according to a genetic code. Proteins catalyze biochemical reactions as enzyme, take part in cell signaling and signal transductions and provide stiffness in cellular structure. Transcription also produces other types of non-protein coding RNA such as transfer RNA, micro RNA, small interfering RNA, ribosomal RNA. All transcription products are crucial for cellular maintenance, growth, proliferation and signaling.

An enzyme called RNA polymerase (RNAP) is responsible for transcription, the process of RNA biosynthesis. RNAP binds at the promoter region in DNA and starts copying DNA

from transcription initiation site forming a complementary RNA. The binding of RNAP is facilitated by TFs. To prevent indiscriminate transcription of DNA, TF enforces selective binding of RNAP through sequential, structural and molecular specificity. The loci, where TFs bind, are TFBSs. Recognizing TFBS is a central task in elucidation of transcription regulatory mechanism. For detail description see [15].

There are different kinds of experimental procedure, e.g. SELEX [9], Dnase [17], F-Seq [1], ChIP-chip and ChIP-Seq [2, 16, 22, 33], to determine TFBSs of a specific TF. Nevertheless, it is costly to perform wet-lab experiments. So great amount of effort is made to detect TFBS computationally. TFBSs are loosely conserved across similar species and sometimes within a single genome. Computational methods take advantage of this conservation and report short, statistically significant, sequences as putative TFBSs. But this approach results in a large amount of false positives. The number of false positives can be reduced by taking a stringent threshold in the predictive models. But again, this approach might miss the actual TFBSs.

Almost every computational method for TFBS prediction has two components: a mathematical model to represent a collection of TFBSs and a search strategy which explores and scores through the search space of different instances of TFBSs to find the likely sequence pattern. The models are also used to capture features of known TFBS collections and to predict putative TFBSs for unannotated sequences. Using appropriate model is crucial for extracting relevant features of a TFBS collection.

There are several TFBS discovery tools with different level of success (see early review in [30]). In another review [26], authors present a layered modeling of the TFBS detection problem. From algorithmic perspective [21] explains different tools. Nevertheless, the performance of a TFBS discovery tools is tightly coupled with the appropriateness of using a model as well as the efficacy of a search procedure. This makes it difficult to trace the effectiveness contributed by certain model alone. To the best of our knowledge, there has been no study until now that assesses the performance of different models used for

prediction of TFBSs . We performed the first large scale study of this type and results are presented in this report. In particular, our contributions are as follows:

1. We compared widely used position weight matrix (PWM) model, fixed-order Markov chain model (MCM), profile hidden Markov model (HMM), pairwise remote dependency model (RDM) and Bayesian network model (BNM).

2. We considered 2–score vector for scoring as well as scalar scoring model for models that allow for that.

3. We developed a computational pipeline and relevant statistics for model assessment

The rest of the thesis is organized as follows. Chapter 2 describes different models that we have assessed. Chapter 3 describes the data, methods and computational implementations. Chapter 4 presents our results and discussions. Chapter 5 contains conclusion and future research directions.

Chapter II

MODEL DESCRIPTIONS

The problem we intend to solve is to assess performance of 12 different models that describe sets of TFBSs. We consider multiple alignments of TFBSs and estimate the parameters of candidate probabilistic models. Then we calculate our defined specificities for different level of sensitivities for individual TFBS sets and background genomic sequences. Depending on the outcomes the models are ranked according to certain performance measures. For an individual TFBS set: (a) we use as inputs multiple alignment of these TFBS sequences and a background sequence; (b) the outcome of assessment for a model are model specificities at different sensitivities and the best parameter setting; (c) when all models are processed, then we rank them according to their corresponding best performance.settings for each model, ranking of models according to their corresponding best performances.

Table II.1 shows a small input example of our aligned TFBSs.

II.1 Model Features

Mathematical modeling is the art of defining a system using the language mathematics. This definition has a lot of benefits. If we can describe a system using mathematics, we can have its precise analysis, can use the standard mathematical operation to manipulate the system, and most importantly we can predict and simulate the system computationally

Table II.1: An example of aligned TFBSs.

Motif	1	2	3	4	5
Motif 1	T	A	T	A	A
Motif 2	T	A	T	A	A
Motif 3	T	A	T	T	A
Motif 4	T	A	T	A	A
Motif 5	G	A	T	A	G
Motif 6	T	A	T	A	A
Motif 7	G	A	T	T	A
Motif 8	G	A	T	A	G

to get the necessary understanding about its behavior.

In our case the system is the set of TFBSs. We model a TFBS using random variables with their interdependencies and a scoring procedure to calculate the degree of match between our defined model and a potential TFBS. In the following we give important criteria for our model choices.

Stochasticity Whether variables are deterministic or probabilistic; we consider only probabilistic variables since they can best capture the uncertainty of protein-DNA binding. Deterministic models such as consensus patterns cannot capture the preference of particular choice.

Dependency Whether variables are independent or dependent of each other; we consider both kinds

Homogeneity Whether variables are position-specific or position-unspecific; we consider only inhomogeneous or position-specific variables because there are evidences that protein-DNA binding is context specific and ordered.

Continuity Whether variables are discrete or continuous. We only consider discrete representation of TFBSs with four character alphabet $\{$A,C,G,T$\}$.

Structure Whether dependency structure has some fixed construct or they are arbitrary. Except for Bayesian networks, dependency structure of other models are predefined.

Bayesian network can learn an arbitrary dependency between TFBS's positions.

Directionality Whether the dependencies are directed or undirected. Our dependencies are undirected in a sense that altering directions of dependencies does not change the outcome.

Dynamics Whether dependencies are dynamic or static; we consider only static dependencies among variable that does not change over time.

Observability Whether the states of the variables are observable; except for profile hidden Markov model, in all of our models all the variables are fully observable.

Order Maximum number of variables that can depend simultaneously on another variable; we consider models of different orders, e.g. zero-order (independent), first-order (pairwise) and higher-order. We also consider both variable-order models, where different variables can have different number of simultaneously dependent variables, and fixed-order models, where all variables have equal number of simultaneously dependent variables. Except for Bayesian networks, we restrict ourselves to fixed-order models.

Adjacency Whether the dependency occurs only between adjacent TFBS's positions. Remote dependency model and Bayesian networks can deal with nonadjacent dependencies.

Parameter Whether the model is parametric or nonparametric. We consider only parametric models. We use categorical probability distribution in tabular format to represent the likelihood of nucleotide features.

Scoring Whether the match score is vector or scalar. The degree of match between a model and a potential TFBS has been predominantly represented by a scalar score/likelihood. We use a score pair or 2-score match model with scores representing critical and noncritical positions/dependencies. We used the concept of information content to distinguish between these positions.

The following sections describe our models and provide examples.

II.2 Position Weight Matrix

PWM is the simplest of all probabilistic models used to represent a family of TFBSs. Early reference of PWM use can be found in [29]. Another important review on PWM can be found in [28]. PWM representation assumes complete independence among the positions in the sequences that are modeled and only captures the position-specific nucleotide distribution. There are arguments both for [7] and against [8] this independence assumption.

Table II.2: A position frequency matrix

Nucleotide	1	2	3	4	5
Adenine (A)	0	8	0	6	6
Cytosine (C)	0	0	0	0	0
Guanine (G)	3	0	0	0	2
Thymine (T)	5	0	8	2	0

To build a PWM our first task is to create a position frequency matrix (PFM) from the alignment of sequences. We need to count the frequencies of A, C, G and T nucleotides at each position. Positions correspond to columns in the matrix. PFM for the Table II.1 is an $\mathcal{R} \times \mathcal{C}$ dimensional matrix where the number of row is $\mathcal{R} = 4$, since we have four possible choices of nucleotide at every position. The number of column is $\mathcal{C} = 5$, since we have 5 positions in the given alignment of sequences. In this model the alignment is very important, since we assume exclusively that nucleotide positions contain information about the structure of TFBSs. Moreover, we normally assume independence among the positional distributions of nucleotides. The resulting PFM for the above example is given in the Table II.2.

To build a probability matrix we need to divide every element by the summation of all the elements from the corresponding column. This gives us the categorical distribution

Table II.3: A probability matrix

Nucleotide	1	2	3	4	5
Adenine (A)	0	1	0	.75	.75
Cytosine (C)	0	0	0	0	0
Guanine (G)	.38	0	0	0	.25
Thymine (T)	.62	0	1	.25	0

of every column. The number of categories in each column is four possible nucleotides. This is a generalization of the Bernoulli distribution where the random variable can have more than two possible outcomes. Using the distribution of Table II.3 we can calculate the likelihood that a given sequence is a putative TFBS using following equation.

$P(x_1 x_2 \dots x_n) = \prod_{i=1}^{n} P(x_i)$

For relatively long sequences the likelihood quickly becomes very small and might cause numeric underflow. For this reason we normally use the log-likelihood which preserve the monotonicity of the likelihood over sequence space but avoids overflow problems. We can calculate the log-likelihood using following equation.

$LL(x_1 x_2 \dots x_n) = \sum_{i=1}^{n} \log P(x_i)$

For example, the likelihood of sequence **TATAA** is $.62 \times 1 \times 1 \times .75 \times .75 = 0.34875$.

We can notice some problems with the probability matrix representation of TFBSs. First, a lot of of entries in that matrix contain zero. This ensures that even a single mismatch in those crucial position will result in no likelihood. But it may not be the case. For example, the sequence **TATAC** and **CCCCC** both will have a likelihood zero in the above scheme. But certainly the former sequence is a much better candidate for a putative TFBS. Having zero entries also presents problem for log-likelihood. Logarithm of zero is infinity for any base. But computer arithmetic cannot deal with true infinity and generates overflow/underflow problem. To avoid this problem, common practice is to add a pseudo-count to each of the position as a prior probability. When no knowledge is known about the background, it is customary to add one to every position which is a Dirichlet's

prior representing complete ignorance. Adding prior avoids the above mentioned computational problems, but the choice of prior itself is a problem. If the number of TFBSs increases, the effect of adding pseudo-counts gradually diminishes. But most of the time the number of total TFBSs is few, and using prior results in a large number of false positives. This essentially ruins the effectiveness of the model. Another problem with the probability matrix is the likelihood is not normalized across different PWM. It is possible to compare the scores of two different sequences generated by same PWM, but it does not make any sense to compare scores generated by different PWMs. The scores might vary considerably even for similar degree of match.

Table II.4: A position weight matrix

Nucleotide	1	2	3	4	5
Adenine (A)	0	.24	0	.18	.18
Cytosine (C)	0	0	0	0	0
Guanine (G)	.09	0	0	0	.06
Thymine (T)	.15	0	.24	.06	0

To avoid the problem of using prior and comparability across different PWMs, we use the additive scoring scheme defined by [4]. This scheme cannot be treated as a likelihood in an exact sense, however it gives an approximation of the degree of similarity. Moreover, the scheme is free from the bias introduced by priors, capable of discarding the putative TFBSs with zero entry if necessary and normalized in such a way that we can get a sense about the degree of similarity for TFBSs across different PWMs. We normalize each entry of PFM using the following equation,

$$PWM_{i,j} = \frac{PFM_{i,j}}{\sum_{k=1}^{n} max(PFM_k)}$$

We sum up the maximum values of all the columns of PFM and divide every elements of PFM with that sum. In our example the maximum values for the 5 positions are $\{5,8,8,6,6\}$ and sum is 33. So we divided all the elements of PFM by 33. The resulting matrix is given in Table II.4. To calculate the match score of an 5-length sequence by PWM, we need to

sum up all the cells of PWM where a match occurs in corresponding position. For example, the sequence **TATAA** has the score .15+.24+.24+.18+.18 = .99 . Ideally for this sequence the score should be exactly 1, since this is the maximum possible match that can occur for the given alignment. But since we restrict ourselves to two digits after decimal there are some round-off errors. For other normalization scheme please see [14].

II.3 Fixed-Order Markov Chain

In TFBS detection, the use of Markov chains [20] and their variants [34] is not new. Markov chains are random processes which possess Markov property. In the simplest terms, Markov property tells that a stochastic process is memoryless and the next state of the governing random variable depends only on the current state of that random variable. This definition was developed for dynamic processes which varies over time. However, Markov chains have been successfully applied to capture the static dependency structure among a system of random variable. Normally we assume an implicit order among the random variables to define the state transition. However, in practice this does not restrict the possible dependency structures.

Markov chains are often represented by conditional probability distribution which intuitively capture the transition directionality. The order of Markov chain depends on how many previous state variables affect the choice for next state variable. If all the state variables are independent of the state of any previous variables, then the Markov chain is of order zero. Our position weight matrix model, discussed in the previous section, can be considered as a zero-order Markov chain, since the variables take values independent of the choices of other variables. If a state variable depends on previous one variable determined by the ordering (temporal or statically assigned) the Markov chain is of first-order. In this way, Markov chain of order two, three and more can be defined. But whatever may be the order, it has to be finite. It is possible to show that we can reformulate any finite-higher-order

Markov chain into first-order Markov chain thus satisfying memoryless property. In our work, we consider Markov chain of order one, two and three. We also keep the order as consistent as possible within the chain, means the order of all the variables is equal with the exception of terminal variable(s) which has no preceding variable(s).

To calculate the probability of a sequence using conditional probability distributions we use the following equation,

$P(x_1x_2 \dots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_2, x_1) \dots P(x_n|x_{n-1}, x_{n-2}, ..., x_1)$

Applying Markov memoryless property, we assume dependence on the last k variables. Then the equation becomes,

$P(x_1x_2 \dots x_n) = P(x_1, \dots, x_k) \prod_{i=k+1}^{n} P(x_i|x_{i-1}, \dots, x_{i-k})$

To avoid the numeric underflow as discussed in previous section we take the log-likelihood. The equation is,

$LL(x_1x_2 \dots x_n) = \log P(x_1, \dots, x_k) + \sum_{i=k+1}^{n} \log P(x_i|x_{i-1}, \dots, x_{i-k})$

Table II.5: A first-order Markov frequency matrix

Neighbor	Position	2 | 1	3 | 2	4 | 3	5 | 4
A	A	0	0	0	4
	C	0	0	0	0
	G	0	0	0	2
	T	0	8	0	0
C	A	0	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
G	A	3	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
T	A	5	0	6	2
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	2	0

To build a Markov conditional probability matrix, our first task is to create a Markov

frequency matrix (MFM) from the alignment matrix. We give an example for a first-order Markov chain. For every position except the first position there is a $4 \times 4 = 16$ cell column. Each cell contains the count of corresponding pair of neighboring nucleotides from the given alignment. Since this is a first-order model we can have maximum four columns for four conditional probability distributions $\mathcal{P}(2 \mid 1)$, $\mathcal{P}(3 \mid 2)$, $\mathcal{P}(4 \mid 3)$, and $\mathcal{P}(5 \mid 4)$. We need to count the frequencies of AA, AC, ... and TT dinucleotides on each column. The resulting Markov frequency matrix for the above example is given in the Table II.5.

Table II.6: A first-order Markov conditional probability matrix

Neighbor	Position	$2 \mid 1$	$3 \mid 2$	$4 \mid 3$	$5 \mid 4$
A	A	0	0	0	.67
	C	0	0	0	0
	G	0	0	0	.33
	T	0	1	0	0
C	A	0	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
G	A	1	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
T	A	1	0	.75	1
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	.25	0

To build a Markov conditional probability matrix we need to divide every element by the summation of all the elements from corresponding 4-cell subcolumn. For example, the third element of the last column is 2 and the summation of its 4-cell subcolumn is $4 + 0 + 2 + 0 = 6$. So in the probability matrix the entry is $2/6 = .33$. This gives us four categorical distribution in every column. The number of categories in each subcolumn is four possible nucleotides. Using the distribution of Table II.6 we can calculate the likelihood that a given sequence is a putative TFBS with the following equation.

$P(x_1 x_2 \ldots x_n) = P(x_1) \prod_{i=2}^{n} P(x_i|x_{i-1})$

Here, $P(x_1)$ can be used from probability matrix built for the previous position specific scoring model. We can calculate the log-likelihood using following equation.

$LL(x_1 x_2 \ldots x_n) = \log P(x_1) + \sum_{i=2}^{n} \log P(x_i|x_{i-1})$

For example, the likelihood of sequence **TATAA** is $.6 \times 1 \times 1 \times .75 \times .67 = 0.3015$.

Table II.7: A first-order Markov weight matrix

Neighbor	Position	1.2	2.3	3.4	4.5
A	A	0	0	0	**.17**
	C	0	0	0	0
	G	0	0	0	.09
	T	0	**.35**	0	0
C	A	0	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
G	A	.13	0	0	0
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	0	0
T	A	**.22**	0	**.26**	.09
	C	0	0	0	0
	G	0	0	0	0
	T	0	0	.09	.00

We can notice that all the problems that were associated with the probability matrix of our previous PWM models also appear here. First, a lot of entries in that matrix contain zero. We discussed that adding prior itself is a problem. Another problem with the probability matrix is the likelihood is not normalized across different Markov probability matrices. To avoid all these problems, we adopt a similar additive scoring scheme here. This scheme also cannot be treated as a likelihood in an exact sense, however it gives an approximation of the degree of similarity with the benefits discussed previously. We normalize each entry of using the following equation,

$MWM_{i,j} = \frac{MFM_{i,j}}{\sum_{k=1}^{n} max(MFM_k)}$

As previous, we sum up the maximum values of all the columns of MFM and divide every elements of MFM with that sum. In our example the maximum values for the 4 dependencies are $\{5,8,6,4\}$ and sum is 23. So we divided all the elements of MFM by 23. The resulting matrix is given in Table II.7. To calculate the match score of an 5-length sequence by our Markov weight matrix (MWM), we need to sum up all the cells of MWM where a match occurs in corresponding dependency. For example, the sequence **TATAA** has the score $.22 + .35 + .26 + .17 = 1$. This is the maximum possible match that can occur for the given alignment. But since we restrict ourselves to two digits after decimal sometimes there are some round-off errors.

II.4 Profile Hidden Markov Model

Hidden Markov models (HMM) are probabilistic model and widely used in many fields including signal processing, speech recognition, bio-informatics etc, including our current problem at hand [13, 32]. One of the powerful capabilities of HMM is that it can capture the potential insertion and deletion events within TFBSs from gapped alignment. It can also accommodate variable-length window size during scanning the genome. The primary difference of HMM with the previously defined Markov chains is that they introduce unobserved or hidden state variables to model the stochastic process.

To define a hidden Markov model completely we need to define follow things:

1. The set of hidden states $S = \{s_1, s_2, \ldots, s_n\}$
2. The output alphabet $A = \{a_1, a_2, \ldots, a_m\}$
3. Probability of being in different states from S initially $\Pi(S) = \{\pi(s_1), \pi(s_2), \ldots, \pi(s_n)\}$
4. Transition probabilities $P = \{p_{ij}\}$ where p_{ij} is the probability of going to state s_j in the next time step given that the current state is s_i. This probability is independent of the previous states which is the Markov property.

Table II.8: An example of gap-aligned TFBSs.

Motif	1	2	3	4	5	6
Motif 1	T		A	T	A	A
Motif 2	T		A	T	A	A
Motif 3	T		A	T	T	A
Motif 4	T		A	T	A	A
Motif 5		G	A	T	A	G
Motif 6	T		A	T	A	A
Motif 7		G	A	T	T	A
Motif 8		G	A	T	A	G
State	M1	I1	M2	M3	M4	M5

5. Output probabilities $O = \{o_s(a)\}$ where $o_s(a)$ is the probability of giving output $a \in A$ given the current state is $s \in S$.

We want to calculate the probability of an observed sequence $o_1 o_2 \dots o_n$ given the model. The following equation is used to find that. It sums up the likelihood of a sequence for all possible state assignments. For a particular state assignment, we can calculate the likelihood using the product of all consecutive state transition probabilities and the emission probabilities of corresponding output. For details of the algorithm please see [23].

$P(o_1 o_2 \dots o_n) = \sum_{\vec{s}} \prod_{i=1}^{n} P(o_i|s_i) \prod_{i=1}^{n} P(s_i|s_{i-1})$

To model TFBS, a specific type of HMM is normally used which is called profile HMM. Here the first task is to create a profile from a given optionally gapped multiple alignment and then estimate all of the above parameters.

A profile can have three kinds of states.

1. 'Match' state where we assume that the corresponding position of the sequence being scanned has a matching with the corresponding column of the given alignment.

2. 'Insert' state where we assume that the corresponding position of the sequence being scanned has an insertion which does not match with our 'Match' model.

3. 'Delete' state where we assume that the corresponding position of the sequence being

scanned does not have a matching with the corresponding column of current 'match' state and this 'match' state can be skipped for future comparison.

Determining the number of 'match' states (or the length of HMM) can be tricky. We can learn the profile using Baum-Welch expectation maximization algorithm. This is computationally expensive but does not guarantee optimal solution. A widely used heuristic to determine 'match' states is to use only those columns which have nucleotides in at least half of the sequences in a gapped alignment. Columns with nucleotides less than half of the sequences can be modeled as 'insert' states. To model deletions with arbitrarily long sequence we can introduce 'delete' states where we do not have any output. We use also 'begin' and 'end' state to denote the start and finish of matching. We should note that the structure of the HMM is fixed before learning parameters in the given scheme.

Table II.9: Transition Matrix for HMM.

State	M1	I1	M2	M3	M4	M5	END
BEGIN	.62	.38	0	0	0	0	0
M1	0	0	1	0	0	0	0
I1	0	0	1	0	0	0	0
M2	0	0	0	1	0	0	0
M3	0	0	0	0	1	0	0
M4	0	0	0	0	0	1	0
M5	0	0	0	0	0	0	1

Table II.10: Emission Matrix for HMM.

States	A	C	G	T
M1	0	0	0	1
I1	0	0	1	0
M2	1	0	0	0
M3	0	0	0	1
M4	.75	0	0	.25
M5	.75	0	.25	0

The output probabilities of 'match' and 'insert' states are calculated from the frequency

of occurrences of each nucleotide in corresponding column of the aligned profile. In Table II.8, the 'match' state 'M4' has 6 'A's and 2 'T's. So the output probability of 'A' in state 'M4' is .75 and output probability of 'T' in state 'M4' is .25 which can be seen in the emission matrix from Table II.10. Prior background probabilities can be used to estimate the 'insert' states output probabilities. However, we only used the frequency from the aligned profile. The transition probability from state i to state j is calculated from the frequency of transitions going to state j while the current state is i. In Table II.8, we can see that in 5 cases there is a transition from 'begin' state to 'M1' state and in 3 cases from 'begin' state to 'I1' state. So the transition probability from 'begin' to 'M1' is 62.5 and from 'begin' to 'I1' is 37.5 which have been reflected in Table II.9. The likelihood of a TFBS is found by summing up the probabilities of all possible state sequences which can generate the given sequence using the above mentioned equation.

II.5 Bayesian Networks

The use of Bayesian networks in TFBS detection can be found in [5] and [6] among others. Bayesian networks are one of the most important types of probabilistic graphical model widely used in diverse areas. Hidden Markov model is a special case of Bayesian network. Probabilistic graphical models are used to represent the dependencies between random variables. In the graph structure the nodes represent random variables and edges represent their dependencies. Edges can be directed or undirected. Where a dependency can be attributed a causal relationship, the direction can be determined. If the causal structure is difficult or impossible to derive but the correlation among the random variables is evident, then the edge is undirected. Absence of edges denote independence. Directed graphical models are called Bayesian network. There are two learning components of Bayesian network, structure and parameter learning. Structure learning deals with learning a directed acyclic graph (DAG) which can best describe the dependencies among random variables.

The parameter learning deals with the estimation of conditional probability distribution among the nodes. A conditional probability table (CPT) states the probabilities of different choices of the 'child' random variable for the given choices of 'parent' random variables. To calculate the likelihood of a given sequence (the values of random variable), there are several inference algorithm. The equation for likelihood inference is as follows,

$$P(x_1x_2 \dots x_n) = P(x_1, \dots, x_k) \prod_{i=k+1}^{n} P(x_i|Pa_{i-1}(x_i), \dots, Pa_{i-k}(x_i))$$

The sequence is of n-length. $Pa(x)$ denotes the parents' of node x. Since the probability sometimes gets very small as discussed in the previous sections, the log-likelihood is used. The equation is as follows,

$$LL(x_1x_2 \dots x_n) = \log P(x_1, \dots, x_k) + \sum_{i=k+1}^{n} \log P(x_i|Pa_{i-1}(x_i), \dots, Pa_{i-k}(x_i))$$

Bayesian networks are powerful models since they can capture arbitrary dependencies. In our experiment, we use the [6] software. We have compared several Bayesian network models with different constraints. The order of a Bayesian network is the number of maximum possible parents that a node can have. Fixed-order Bayesian network is where all the nodes have equal number of parents except the root nodes. We compare fixed-order Bayesian network with order one, two and three. We also use general Bayesian network with arbitrary structure. Since learning the optimal structure for a Bayesian network is an NP-hard problem, we use genetic algorithm as an approximate search procedure to explore the solution space for a local optimal as implemented by [6].

Table II.11: Prior Probability for Bayesian network

Nucleotide	$\mathcal{P}(1)$
Adenine (A)	0
Cytosine (C)	0
Guanine (G)	.38
Thymine (T)	.62

For our given alignment in Table II.1 let us assume the directed dependencies are as

following, $1 \rightarrow 2$, $1 \rightarrow 3$, $1 \rightarrow 5$, $2 \rightarrow 3$, $2 \rightarrow 4$, $3 \rightarrow 4$ and $4 \rightarrow 5$. The corresponding conditional probability tables (CPT) are $P(2 \mid 1)$, $P(3 \mid 2,1)$, $P(4 \mid 3,2)$, $P(5 \mid 4,1)$. This is an example of second-order Bayesian network where the maximum number of parents that a node can have is two. We also need the prior probability for the root nodes to compute the likelihood. In this example the root is node one. The prior probability table for root node is given in Table II.11. This gives the distribution of nucleotides in position one whose calculation process is similar to that of position weight matrix.

Table II.12: Conditional probability table for $\mathcal{P}(2 \mid 1)$

Nucleotide	A	C	G	T
Adenine (A)	0	0	0	0
Cytosine (C)	0	0	0	0
Guanine (G)	1	0	0	0
Thymine (T)	1	0	0	0

The CPT's are calculated from the nucleotide frequencies given by the alignment in Table II.1. For example in Table II.12 the nucleotides at independent position (position 1) constitutes the rows and the nucleotides at dependent position (position 2) constitutes the columns. Since whenever we have a 'G' in position 1, we have a 'A' in position 2. So the conditional probability $P(x_2 = A \mid x_1 = G) = 1$ which is reflected in Table II.12.

To clarify we give another example of CPT. The Table II.13 represent a second order CPT which capture the dependencies between position 5, 4, and 1. Since the position 5 has two parents , 4 and 1, the dependency is of order two. The CPT is calculated from the nucleotide frequencies given by the alignment in Table II.1. In Table II.13 the nucleotides at independent positions (position 4 and 1) constitutes the rows of dinucleotides and the nucleotides at dependent position (position 5) constitutes the columns. Since whenever we have a 'T' in position 4 and a 'T' in position 1, we have a 'A' in position 5. So the conditional probability $P(x_5 = A \mid x_4 = T, x_1 = T) = 1$ which is reflected in Table II.13.

To calculate the match score of a sequence by Bayesian network, we need to multiply

Table II.13: Conditional probability table for $\mathcal{P}(5 \mid 4, 1)$

Nucleotide	A	C	G	T
A.A	0	0	0	0
A.C	0	0	0	0
A.G	0	0	1	0
A.T	1	0	0	0
C.A	0	0	0	0
C.C	0	0	0	0
C.G	0	0	0	0
C.T	0	0	0	0
G.A	0	0	0	0
G.C	0	0	0	0
G.G	0	0	0	0
G.T	0	0	0	0
T.A	0	0	0	0
T.C	0	0	0	0
T.G	1	0	0	0
T.T	1	0	0	0

all the relevant entries from CPT's and prior tables. For example, the sequence **TATAA** has the score $P(T) \times P(A \mid T) \times P(T \mid A, T) \times P(A \mid T, A) \times P(A \mid A, T) = .625 \times 1 \times 1 \times .75 \times 1 = 0.46875$. This is the maximum possible match that can occur for the given alignment.

II.6 Pairwise Remote Dependency Model

Generalized dinucleotide models can be found in the work [27]. This pairwise remote dependency model considers all possible two-way dependencies among the variables. The number of parameters is quadratic to the length of TFBSs. This is a highly specific model which can capture subtle relations between remote positions quite accurately. The downside is there is a risk of over-fitting if the training set is not a nearly accurate representation of TFBS profile or the training set is so small that the coverage of true positives is not adequate. But recent experimental techniques are overcoming the problems due to low

coverage and sample inadequacy. Modern procedures such as ChIP-seq can generate genome-wide comprehensive TFBS samples in large amount to be used by highly specific models.

Table II.14: A dinucleotide frequency matrix

Pair	1.2	1.3	1.4	1.5	2.3	2.4	2.5	3.4	3.5	4.5
A.A	0	0	0	0	0	6	6	0	0	4
A.C	0	0	0	0	0	0	0	0	0	0
A.G	0	0	0	0	0	0	2	0	0	2
A.T	0	0	0	0	8	2	0	0	0	0
C.A	0	0	0	0	0	0	0	0	0	0
C.C	0	0	0	0	0	0	0	0	0	0
C.G	0	0	0	0	0	0	0	0	0	0
C.T	0	0	0	0	0	0	0	0	0	0
G.A	3	0	2	1	0	0	0	0	0	0
G.C	0	0	0	0	0	0	0	0	0	0
G.G	0	0	0	2	0	0	0	0	0	0
G.T	0	3	1	0	0	0	0	0	0	0
T.A	5	0	4	5	0	0	0	6	6	2
T.C	0	0	0	0	0	0	0	0	0	0
T.G	0	0	0	0	0	0	0	0	2	0
T.T	0	5	1	0	0	0	0	2	0	0

To calculate a pairwise dinucleotide weight matrix, our first task is to create a dinucleotide frequency matrix (DFM) from the alignment matrix. We need to count the frequencies of AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, and TT dinucleotides between each pair of columns, (1,2), (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4), (3,5), and (4,5). DFM for the Table II.1 is an $\mathcal{R} \times \mathcal{C}$ dimensional matrix where the number of row is $\mathcal{R} = 16$, since we have sixteen possible choices of nucleotide for every pair of column. The number of column is $\mathcal{C} = 10$, because we have 5 positions in the given alignment matrix from which we can have 10 possible pair of non-duplicating columns. In this model the pairing is very important, since we assume exclusively that nucleotide pairings contain information about the structure of TFBSs. The resulting DFM for the Table II.1 is given in the Table II.14.

There is no straightforward way to calculate the likelihood of a sequence from DFM.

Table II.15: A dinucleotide weight matrix

Pair	1.2	1.3	1.4	1.5	2.3	2.4	2.5	3.4	3.5	4.5
A.A	0	0	0	0	0	**.11**	**.11**	0	0	**.07**
A.C	0	0	0	0	0	0	0	0	0	0
A.G	0	0	0	0	0	0	.04	0	0	.04
A.T	0	0	0	0	**.15**	.04	0	0	0	0
C.A	0	0	0	0	0	0	0	0	0	0
C.C	0	0	0	0	0	0	0	0	0	0
C.G	0	0	0	0	0	0	0	0	0	0
C.T	0	0	0	0	0	0	0	0	0	0
G.A	.05	0	.04	.02	0	0	0	0	0	0
G.C	0	0	0	0	0	0	0	0	0	0
G.G	0	0	0	.04	0	0	0	0	0	0
G.T	0	.05	.02	0	0	0	0	0	0	0
T.A	**.09**	0	**.07**	**.09**	0	0	0	**.11**	**.11**	.04
T.C	0	0	0	0	0	0	0	0	0	0
T.G	0	0	0	0	0	0	0	0	.04	0
T.T	0	**.09**	.02	0	0	0	0	.04	0	0

To bulid a dinucleotide weight matrix (DWM) for that [27] use the ordinary position weight matrix as prior to every cell of the DFM. The prior is integrated using following equation,

$DWM_{i,j}[a,b] = \frac{DFM_{i,j}[a,b]+16\times PWM_i[a]\times PWM_j[b]}{\sum_{p=\{A,C,G,T\}}\sum_{q=\{A,C,G,T\}} DFM_{i,j}[p,q]+16}$

Here $DWM_{i,j}[a,b]$ is the cell of pairwise remote dependency weight matrix which is associated positions i and j as well as nucleotide a and b. $PWM_i[a]$ is the cell of position weight matrix associated with position i and nucleotide a. PWMs are calculated using previously described procedure. [27] use the following equation to calculate the approximate likelihood of an n-length sequence.

$P(x_1x_2\ldots x_n) = \prod_{i=1}^{n} P(x_i|x_1,x_2,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = \prod_{i=1}^{n} P(x_i|\{x_{j\neq i}\})$

The quantity $P(x_i|x_{j\neq i})$ is estimated by the following Bayesian expression,

$P(x_i|\{x_{j\neq i}\}) = \frac{P(\{x_{j\neq i}\}|x_i)P(x_i)}{\sum_{q=\{A,C,G,T\}} P(\{x_{j\neq i}\}|x_i=q)P(x_i=q)}$

The quantity $P(\{x_{j\neq i}\}|x_i)$ is further approximated by

$P(\{x_{j\neq i}\}|x_i) = \prod_{j=1(j\neq i)}^{n} P(x_j|x_i) = \prod_{j=1(j\neq i)}^{n} \frac{P(x_j,x_i)}{P(x_i)}$

[27] use $DWM_{i,j}[x_i,x_j]$ in place of $P(x_j,x_i)$ and $PWM_i[x_i]$ in place of $P(x_i)$.

Here also, we notice that all the problems that was associated with the probability matrix of our previous PWM or MWM models also appear here. First, a lot of entries in the matrix may have zero value which can give zero likelihoods for both nearly likely and far apart sequences. Although the scheme uses PWM as a better estimation for priors, calculating PWM has its own problems. The problem of the probability matrix being not normalized is still there. As an added problem, the featured scheme is computationally costly for massive matching due to a complicated approximate likelihood estimation. To avoid all these problems, we adopt a simple additive scoring scheme as before. This scheme also cannot be treated as a likelihood in an exact sense (no exact scheme for calculating likelihood for DWM is known yet), however it gives an approximation of the degree of similarity with the benefits discussed previously. We normalize each entry of using the following equation from [3],

$DWM_{i,j}[a,b] = \frac{DFM_{i,j}[a,b]}{\sum_{i<j} max(DFM_{i,j})}$

We sum up the maximum values of all the columns of DFM and divide every elements of DFM with that sum. In our example the maximum values for the 10 dependencies are $\{$5,5,4,5,8,6,6,6,6,4$\}$ and sum is 55. So we divided all the elements of DFM by 55. The resulting matrix is given in Table II.15. To calculate the match score of an 5-length sequence by our DWM, we need to sum up all the cells of DWM where a match occurs in corresponding dependency. For example, the sequence **TATAA** has the score $..09 + .09 + .07 + .09 + .15 + .11 + .11 + .11 + .11 + .07 = 1$. This is the maximum possible match that can occur for the given alignment.

II.7 Scoring Model

In this study, we extend the idea of importance or criticality of a position [18]. Although the idea can be extended to a general version, for simplicity we restrain ourself to two sets of positions, critical and noncritical. This is a special case of vector/multidimensional scoring

where the score/likelihood of an input sequence is represented by a vector of scores rather than a single scalar score. Among the above models, for PWM, MCM and pairwise RDM, we use two different scoring schemes. We use a single scalar score and a two–score vector to compute the likelihood of matching.

Each of the three models has a weight matrix from which we show in the previous sections with examples how we calculate the likelihood. The score $S_{scalar} = \sum_{i=1}^{l} WM_i[j]$ where the weight matrix (WM) can be a PWM, MWM or DWM. The number of column in WM is l and j is the appropriate row for nucleotide or a combination of nucleotides corresponding to ith column. To compute the two-score vector we separate all the columns of WM into two groups, critical and noncritical. To compute critical and noncritical columns we calculate the information content (IC) of every column which denotes the level of conservation and in our terms the level of criticality of that column. We use the equation $IC_i = \log_2(m) + \sum_{j=1}^{m} WM_i[j] * \log_2(WM_i[j])$ to calculate the information content of ith column where m is the total number of nucleotide or combination of nucleotides available for that column. More simply it is the total number of rows. In this calculation we should take care when $WM_i[j] = 0$, since $\log_2(WM_i[j])$ is undefined. We put zero for those terms since it is also a factor of that term. When we have information contents of all the column available, we can calculate a suitable central tendency. In our case it was median. All the columns with information content equal or greater than median are marked as critical and the rest of the columns are marked as noncritical. So the critical score $S_{crit} = \sum_{(1 \le i \le l) \wedge IC_i \ge median(IC)} WM_i[j]$ and the noncritical score $S_{noncrit} = \sum_{(1 \le i \le l) \wedge IC_i < median(IC)} WM_i[j]$. It gives a two-score vector $\bar{S} = \{S_{crit}, S_{noncrit}\}$.

Figure II.1 shows the dependency structure of our single scalar score models. Figure II.2 shows the dependency structure of our two-score vector models. Finally Figure II.3 shows the dependency structure of our Bayesian network and hidden Markov model. The position specific random variables are denoted using circles and dependency between two nodes is shown using a directed edge.

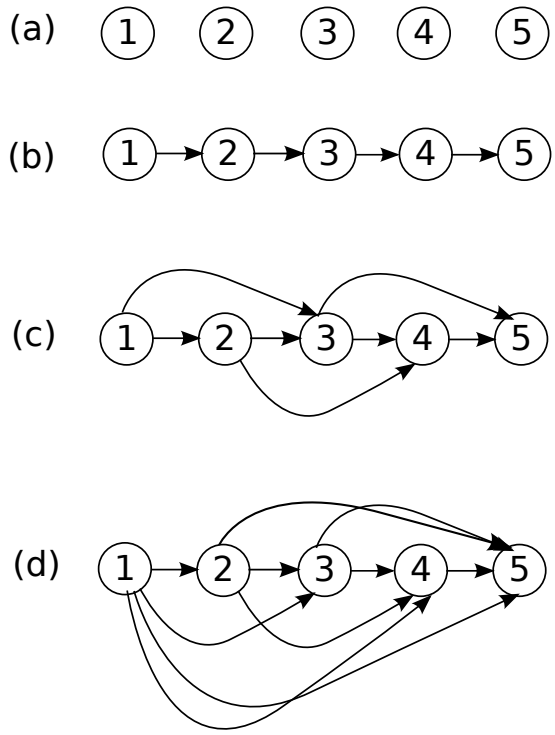

Figure II.1: Dependency structure of 1-score models

Figure II.1(a) shows the absence of any dependency in position weight matrix model giving no edges. Figure II.1(b) shows a first-order Markov dependency in a Markov chain model by giving one single edge to every node except the first node. Figure II.1(c) shows a second-order Markov dependency in a Markov chain model by giving two edges to every node except the first two nodes. Figure II.1(d) shows all possible pairwise dependency in a remote dependency model by giving a single edge between all possible pair of nodes. Single score model is represented by keeping all node circles similar in every figure.

We give a corresponding two-score representation in Figure II.2. Figure II.2(a) shows that position two and four are critical in PWM models by using the borders of the circles thick. Figure II.2(b) shows that Markov dependencies $2 \rightarrow 3$ and $4 \rightarrow 5$ are the critical dependencies in the Markov chain model by making those two edges thick. Figure II.2(c)

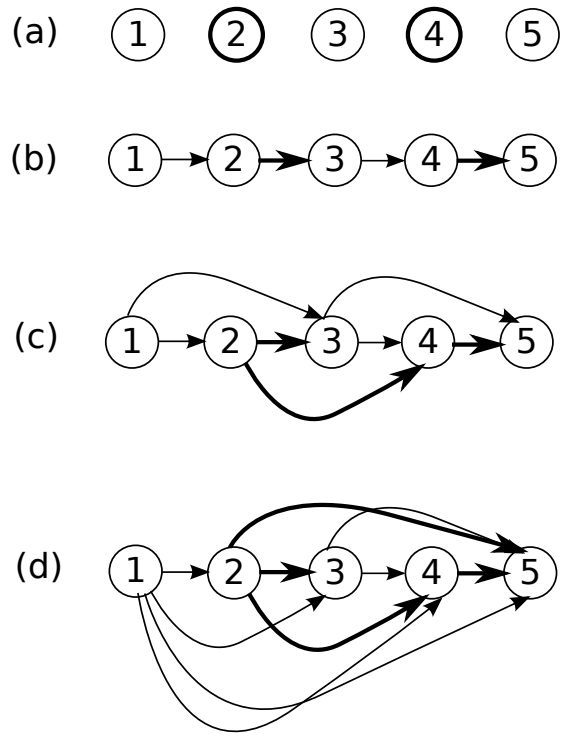

Figure II.2: Dependency structure of 2-score models

shows that Markov dependencies $2 \rightarrow 3$, $2 \rightarrow 4$ and $4 \rightarrow 5$ are the critical dependencies in the Markov chain model. Figure II.2(d) shows that the pairwise dependencies $2 \rightarrow 3$, $2 \rightarrow 4$, $2 \rightarrow 5$ and $4 \rightarrow 5$ are the critical dependencies of the remote dependency model. Noncritical positions and dependencies are represented using thin strokes.

Figure II.3(a), II.3(b) and II.3(c) shows a first-order, second-order and variable-order Bayesian network respectively. We can see that the PWM model and MCM are special cases of Bayesian network. But this is not true for pairwise remote dependency matrix model. Figure II.3(d) shows the basic structure of hidden Markov model. Here, the start and end squares are BEGIN and END states respectively. States 1 through 5 are the hidden MATCH states. The output variables are represented by $\{O_i\}$. The bottom squares represent INSERT or DELETE states. A typical dependency structure is shown

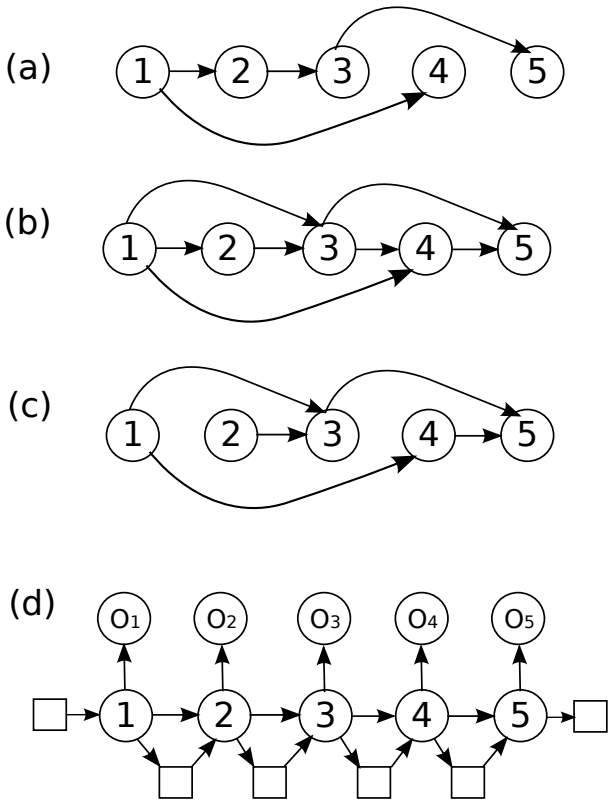

Figure II.3: Dependency structure of Bayesian network and hidden Markov model

using directed edges. Note that the given model is of first-order.

Chapter III

METHODS AND IMPLEMENTATIONS

III.1 Data Preparation

Our primary TFBS data sources are TRANSFAC [31] and JASPAR [25] databases. We have used the commercial version of release 2011 for TRANSFAC. We extract all the TFBS alignments from both databases. We did not use PWMs, but rather the raw TFBS sequences. We keep those alignments done by databases where all sequences are of equal length and do not contain any gap or unknown character. If a default alignment does not fulfill the mentioned criteria, we reprocess them. We look at the leftmost column of the alignment. If less than 10% of the sequences (or only one for sets with less than 10 sequences) have terminal gaps, we exclude those sequences. Otherwise we crop leftmost letter in other sequences and repeat the previous step. We stop when we have a good alignment for the left side. We do the same thing for the right side. Then if less than 10% of the remaining sequences (or only one for sets with less than 10 sequences) have unknown character(s) or gap(s) in the middle, we exclude those sequences. If less than 10% of the sequences (or only 1 for sets less than 10 sequences) have letter(s) in the middle causing

gaps in all other sequences, than we exclude them also. If the final sets have less than 5 sequences or length two time shorter or less than the initial alignment length, than we discard that alignment.

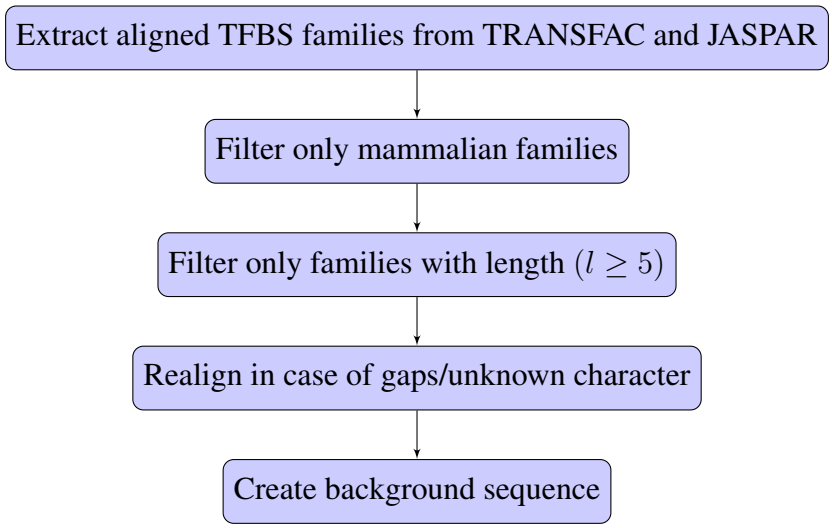

Figure III.1: Work–flow for data preparation

We build our background sequence from human genome version hg19 . The sequences are collected from public sources such as UCSC Genome Browser [24]. To built a random background of 100 Mb (3% of total human genome) we took equally spaced (in terms of genomic co-ordinates) chunks from all the chromosomes. This background is used as our assumed negative samples. The summarized Work–flow is given in Figure III.1.

III.2 Scalar Scoring

To learn the parameters for PWM model, MCM of order two, three and four as well as pairwise RDM with a single scalar scoring scheme we follow a similar procedure.

Calculating weight matrix

Our first task is to create frequency matrices (FM) from the alignment of sequences. For PWM, we need to count the frequencies of A, C, G and T nucleotides on each column of

alignment. For RDM, the frequencies of AA, AC,... TT dinucleotides between each pair of columns need to be calculated. For Markov chains, the length of nucleotide combinations depend on the order of dependency. For first-order dependency we need the frequencies of AA, AC,... TT dinucleotides between the pair of corresponding columns. For second and third order the frequencies for all the trinucleotides and tetranucleotides among corresponding triplets and quadruplets respectively. We normalize all FM using the procedure mentioned in the previous chapter to create weight matrix (WM) using the following equation,

$WM_{i,j} = \frac{FM_{i,j}}{\sum_{k=1}^{n} max(FM_{-,k})}$

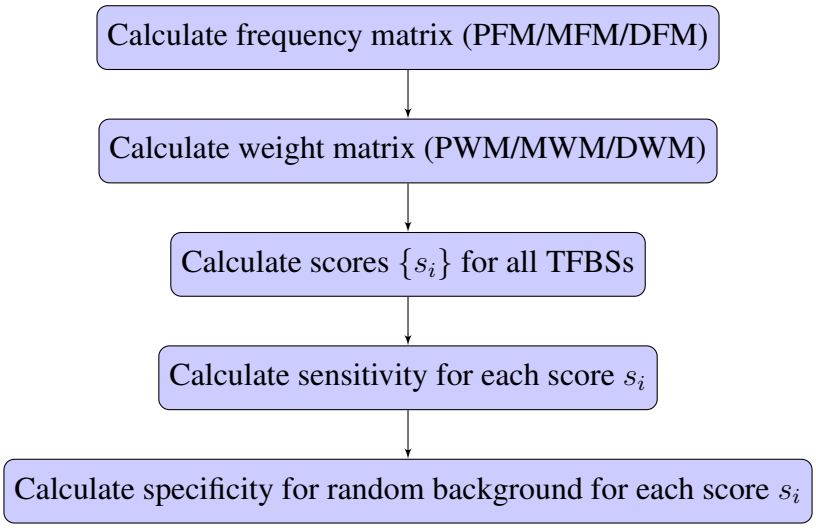

Figure III.2: Work–flow for single scalar score models

Calculating TFBS scores

Next we calculate the score for each TFBS from the alignments by the procedure described in the previous chapter. To calculate score of a sequence $x_1x_2 \ldots x_n$ by PWM we use the following equation. The function $row(x_i)$ returns the correct row-index in PWM for the nucleotide x_i.

$S(x_1x_2 \ldots x_n) = \sum_{i=1}^{n} PWM_{row(x_i),i}$

With RDM we use the following equation. The function $row(x_i, x_j)$ returns the correct

row-index in DWM for the nucleotide pair (x_i, x_j) and function $col(i, j)$ returns the correct column-index in DWM for the corresponding ith and jth columns.

$S(x_1x_2 \ldots x_n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{j=n} DWM_{row(x_i,x_j),col(i,j)}$

With MCM of order k (k-MCM) we use the following equation. The function $row(x_i, x_{i+1}, \ldots, x_{i+k-1})$ returns the correct row-index in MWM of order k (k-MWM) for the nucleotide k-tuple $(x_i, x_{i+1}, \ldots, x_{i+k-1})$.

$S(x_1x_2 \ldots x_n) = \sum_{i=1}^{n-k+1} kMWM_{row(x_i,x_{i+1},\ldots,x_{i+k-1}),i}$

Calculating sensitivity

Suppose we have m TFBSs and the set of matching scores for them is $S = \{s_1, s_2, \ldots, s_m\}$. We calculate the set U of t unique scores from this set which is $U = \{u_1, u_2, \ldots, u_t\}$. Then we calculate the sensitivity for every u_i. For a particular score u_i, we identify the b_i number of TFBS whose scores are greater or equal to u_i. The sensitivity we can achieve with a threshold u_i is equal to $\frac{b_i}{m} \times 100$. We calculated sensitivity for all the u_i.

Calculating specificity

Next we calculate the specificity for every threshold u_i. For this we use our random background from the data preparation phase. We slide a scanning window of length C, the number of column in the alignment, over the background sequence and calculate the score of that window at every position. We identify those positions or hits, where the score of the window is greater than or equal to threshold u_i. The higher is the number of hits, the lower is the specificity. We calculated specificity for all u_i. The summarized Work–flow is given in Figure III.2.

III.3 Two-vector Scoring

To learn the parameters for PWM model, MCM of order two, three and four (2/3/4-MCM) as well as pairwise RDM with a two-vector scoring scheme we follow a procedure similar to the one from the previous section. The first step is to create FMs and WMs. We skip the discussion on them since they are the same to that of single scalar scoring. The next steps are described below.

Calculating information content

Information content (IC) is a measure to indicate the level of nucleotide conservation in the alignments. If the information content of a column (a position in case of PFM and a dependency in case of others) is high then that column is likely to be crucial in the model. We capture the notion of information content through the concept of entropy used in information theory. Entropy is a measure of randomness. The more random the data is, the more entropy it has. We calculate information content by subtracting the entropy of a column from the maximum possible information content of that column. The maximum possible information content in our case is $\log_2 4 = 2$ bits for PFM, since we have 4 choices for every position of our alignment and that can be encoded by two bits in binary representation. In the case where matrix rows indicate dinucleotide, the maximum possible information content is $\log_2 16 = 4$ bits. We have used the following equation for information content of ith column,

$IC_j = \log_2 R + \sum_{i=1}^{4^{k+1}} \frac{FM_{i,j}}{R} \times \log_2(\frac{FM_{i,j}}{R})$

Here R is the total number of rows in alignment. The dependency order is k which means we have 4^{k+1} rows in our FM.

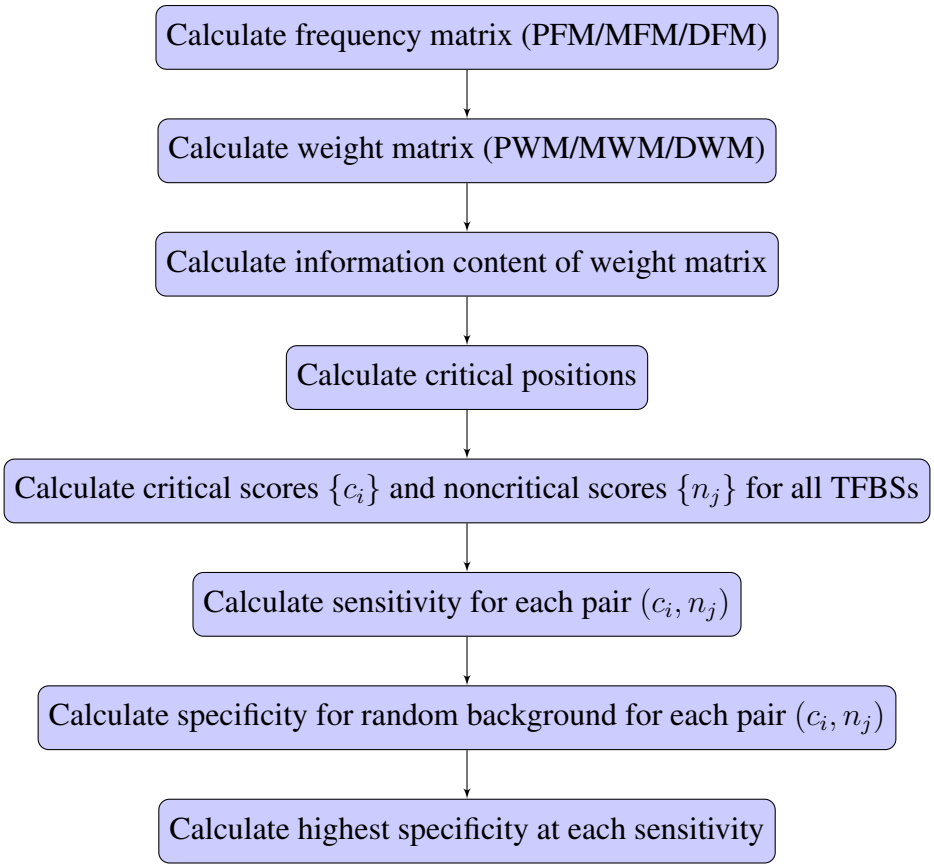

Figure III.3: Work–flow for two-score vector models

Calculating critical position

After calculating the information content we make a two-way division of all the positions/dependencies namely critical positions/dependencies and noncritical positions/dependencies. We calculate a threshold for information content. If a certain position/dependency (a column in frequency matrix) has an equal or higher information content than the threshold, we will refer to it as critical. All other columns are assumed noncritical. For our task, we take the median of all the information content as the threshold.

Calculating critical and noncritical scores for TFBSs

Next we calculate critical and noncritical scores for each TFBS from the alignment in a similar way to that of the scalar score case. We calculate critical score S_c and noncritical score S_n of a sequence $x_1x_2 \ldots x_n$ by PWM as follows. The function $row(x_i)$ returns the correct row-index in PWM for the nucleotide x_i.

$S_c(x_1x_2 \ldots x_n) = \sum_{1 \leq i \leq n \wedge IC_i \geq med(IC)} PWM_{row(x_i),i}$

and,

$S_n(x_1x_2 \ldots x_n) = \sum_{1 \leq i \leq n \wedge IC_i < med(IC)} PWM_{row(x_i),i}$

Here IC_i is the information content of ith column and $med(IC)$ returns the median of the column information contents. The case for RDM is as follows. The function $row(x_i, x_j)$ returns the correct row-index in DWM for the nucleotide pair (x_i, x_j) and function $col(i, j)$ returns the correct column-index in DWM for the corresponding ith and jth columns.

$S_c(x_1x_2 \ldots x_n) = \sum_{i=1}^{n-1} \sum_{i+1 \leq j \leq n \wedge IC_{col(i,j)} \geq med(IC)} DWM_{row(x_i,x_j),col(i,j)}$

and,

$S_n(x_1x_2 \ldots x_n) = \sum_{i=1}^{n-1} \sum_{i+1 \leq j \leq n \wedge IC_{col(i,j)} < med(IC)} DWM_{row(x_i,x_j),col(i,j)}$

With k-MCM we use the following equation. The function $row(x_i, x_{i+1}, \ldots, x_{i+k-1})$ returns the correct row-index in k-MWM for the nucleotide k-tuple $(x_i, x_{i+1}, \ldots, x_{i+k-1})$.ss

$S_c(x_1x_2 \ldots x_n) = \sum_{1 \leq i \leq n-k+1 \wedge IC_i \geq med(IC)} kMWM_{row(x_i,x_{i+1},\ldots,x_{i+k-1}),i}$

and,

$S_n(x_1x_2 \ldots x_n) = \sum_{1 \leq i \leq n-k+1 \wedge IC_i < med(IC)} kMWM_{row(x_i,x_{i+1},\ldots,x_{i+k-1}),i}$

Calculating sensitivity

Let us have m TFBSs. Let the set of critical scores for them be $CS = \{cs_1, cs_2, \ldots, cs_m\}$ and the set of noncritical scores for them be $NS = \{ns_1, ns_2, \ldots, ns_m\}$. We calculate the set CU of t_c unique critical scores from this set which is $CU = \{cu_1, cu_2, \ldots, cu_{t_c}\}$.

We also calculate the set NU of t_n unique noncritical scores from this set which is $NU = \{nu_1, nu_2, \ldots, nu_{t_n}\}$. Then we calculate the sensitivity for every pair (cu_i, nu_j). For a particular score pair (cu_i, nu_j), we identify the b_i number of TFBSs whose critical scores are greater than or equal to cu_i and noncritical scores are greater than or equal to nu_j. The sensitivity we can achieve with a threshold pair (cu_i, nu_j) is equal to $\frac{b_i}{m} \times 100$. We calculate sensitivity for all the pair (cu_i, nu_j).

Calculating specificity

Next we calculate the specificity for every threshold pair (cu_i, nu_j). For this we use our random background from the data preparation phase. We slide a window of length C, the number of column in the alignment, over the background sequence and calculate the critical and noncritical score of that window at every position. We identify those positions or hits, where the score of the window is greater or equal to threshold u_i. The higher is the number of hits, the lower is the specificity. We calculated specificity for all u_i. The summarized Work–flow is given in Figure III.3.

Highest Specificity at Each Sensitivity

Now we have a list of sensitivity and specificity pairs versus corresponding pair of score. At this stage it may be possible to have different specificity for a single sensitivity corresponding to different pair of scores. We calculate the highest specificity that can be achieved for each sensitivity. So, we scan through the list and make another list of sensitivity versus highest specificity and the corresponding score pair.

III.4 Building Hidden Markov Model

For hidden Markov model, we take aligned TFBS sequences as input. Then we label each position of the sequences with 'MATCH' state IDs. Then we calculate the transition matrix

and the emission matrix of the model similar to the way by which we build frequency matrices. The transition matrix gives us the probability of transitions between two states. For a particular pair of states (s_i, s_j), we count the total number of TFBSs where a transition from s_i to s_j happens. Dividing with the total number of TFBSs give us the transition probability $p(s_i, s_j)$. Emission matrix gives us the probability of getting A,C,G and T in each state. Calculating the emission probability is similar to the calculation of the probabilities in a PWM column. In our case, if a position is empty in more than half of the sequences we label the position as an insertion state, otherwise we label it as a matching state.

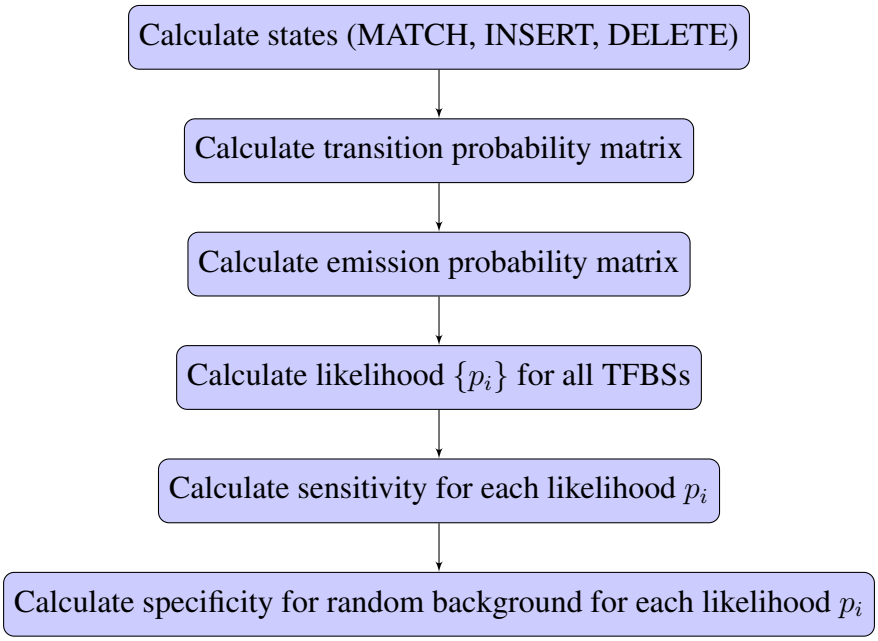

Figure III.4: Work–flow for hidden Markov model

When we have the transition matrix and the emission matrix we can calculate the probability of TFBS sequences by forward algorithm. We call that probability as score of the sequence. Forward algorithm is a simple and efficient dynamic algorithm. With this we compute the scores/probabilities of every TFBSs. Calculating sensitivity and specificity for hidden Markov model is similar to that of single scalar scoring model. The summarized Work–flow is given in Figure III.4.

III.5 Building Bayesian Network

For a Bayesian network model, we use the existing VOBN software [6]. This program has options for both homogeneous model which is not position specific and inhomogeneous model which is position specific. We use inhomogeneous model. Then the program takes the input/output file names and the nucleotide characters. It also takes the order and constraints of dependency. The program build a Bayesian network model from data and can also evaluate score for given sequences. As previously, we calculate score/likelihood for all TFBSs from the alignment.

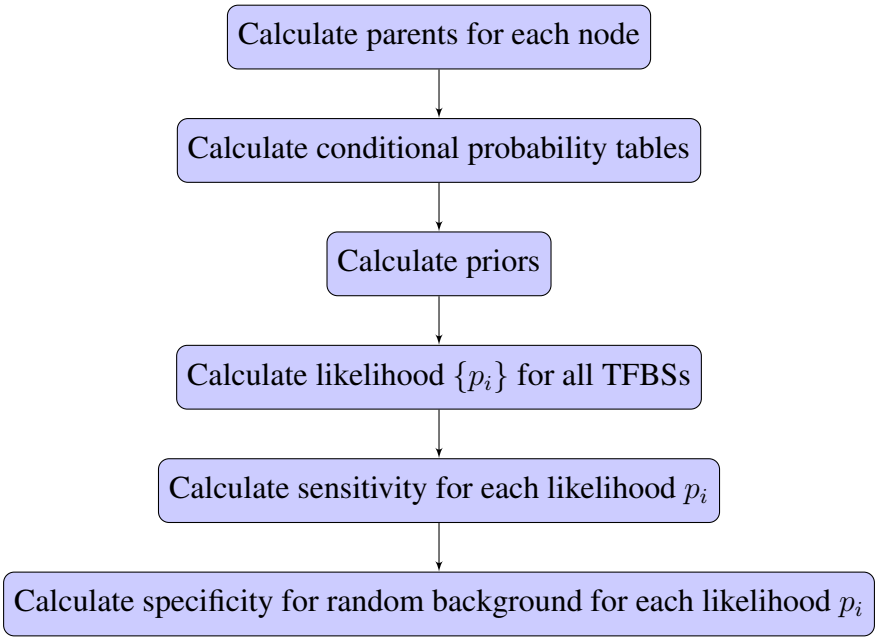

Figure III.5: Work–flow for Bayesian network

Calculating sensitivity and specificity for BNM is similar to that of single scalar scoring model. The summarized Work–flow is given in Figure III.5.

III.6 Prediction and Validation

For every model and TFBS alignment, at specific sensitivity we have a corresponding threshold (learned from the random background) and specificity. Depending on the sensitivity, we can find related threshold and can scan through any sequence to find putative TFBSs. The lower is the sensitivity, the more stringent/higher is the threshold and the less number of sites will be predicted. We need to strike a balance between sensitivity and specificity to get a reasonable amount of predictions. In our comparisons, we decided to operate on at least 90% sensitivity level. At that level if the number of hit is less than one in 100000 nucleotides, we take it as sufficiently specific and we look for the highest sensitivity for which it maintains the above-mentioned specificity. But if the number of hit is more than one in 100000 nucleotides, we do not loose the stringency for sensitivity less than 90%. In this way for each model and alignment pair we get a threshold. We assume that these thresholds can be used for relatively sensitive predictions of TFBSs in unknown sequences with adequate specificity.

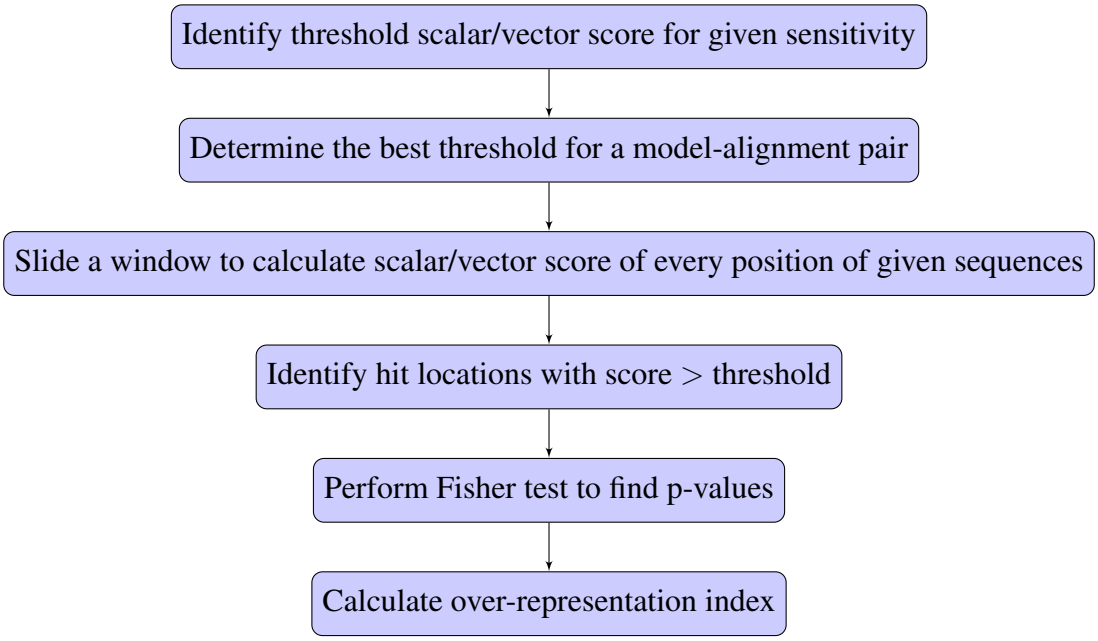

Figure III.6: Work–flow for prediction/validation

Chapter IV

RESULTS AND DISCUSSIONS

We use a total of 587 alignments of TFBSs for more than 100 TFs from both TRANSFAC and JASPAR. Among them 381 matrices from TRANSFAC are taken as they are, since their alignments are good. We took a total of 104 alignment matrices from JASPAR most of which have been taken as they are. Finally we reprocessed 102 alignments from TRANSFAC in the data preparation step to realign or handle gaps. Among these we choose 10 sample alignments. Detail results are available on a separate spreadsheet. Of these 10 matrices, 5 matrices have a length < 10 nucleotides (ETS1, YY1, SPI1, USF1, and E2F1) and 5 matrices have a length ≥ 10 nucleotides (MAX, MEF2A, NFKB1, SRF, and STAT1).

IV.1 Sensitivity Analysis

In our comparison, we choose to maintain at least 90% sensitivity level. The actual sensitivity is determined using the method given in the previous chapter. The point to note is that for longer alignments and more specific models (such as RDM), sensitivity is higher. From the sensitivity context, pairwise RDM outperforms other models.

Table IV.1 and Table IV.2 are samples of our sensitivity results for TFBSs of length less than and greater than (or equal to) ten nucleotides respectively. We can see a consistent sensitivity level of above than 90% wh ch is our chosen sensitivity level.

Table IV.1: Sensitivity for length < 10 nucleotides

Model	Score	ETS1	YY1	SPI1	USF1	E2F1
PWM	1S	92.5	94.12	90.48	90	100
	2S	92.5	94.12	90.48	90	100
2–MER	1S	92.5	94.12	90.48	90	100
	2S	90	94.12	90.48	90	90
3–MER	1S	92.5	94.12	90.48	90	100
	2S	90	94.12	90.48	90	100
4–MER	1S	92.5	94.12	90.48	90	100
	2S	90	94.12	90.48	90	100
RDM	1S	92.5	94.12	90.48	90	100
	2S	90	94.12	90.48	90	100
HMM	1S	92.5	94.12	90.48	90	100
BN	1S	90	94.12	90.48	90	90

Table IV.2: Sensitivity for length ≥ 10 nucleotides

Model	Score	MAX	MEF2A	NFKB1	SRF	STAT1
PWM	1S	94.12	91.38	94.44	95.65	93.33
	2S	94.12	91.38	94.44	93.48	93.33
2–MER	1S	94.12	91.38	94.44	95.65	100
	2S	94.12	91.38	94.44	93.48	100
3–MER	1S	94.12	91.38	100	95.65	100
	2S	94.12	91.38	100	95.65	100
4–MER	1S	94.12	91.38	100	100	100
	2S	94.12	91.38	100	100	100
RDM	1S	100	91.38	100	100	100
	2S	100	91.38	100	100	100
HMM	1S	94.12	91.38	94.44	95.65	93.33
BN	1S	94.12	93.1	94.44	91.3	93.33

IV.2 Specificity Analysis

The specificities corresponding to the above sensitivities are determined from the total number of hits on the random background created from human genome assuming all the TFBS occurrences are false positives there. The point to note is that for longer alignments and more specific models (such as RDM), the number of hits decreases (specificity increases).

From the specificity context, pairwise RDM and MCM of higher order outperform other models. The specificity can also be expressed by the average gap length between the two spurious hits.

Table IV.3: Number of hits for length < 10 nucleotides

Model	Score	ETS1	YY1	SPI1	USF1	E2F1
PWM	1S	681673	441835	5425761	20383	792
	2S	681673	441835	4640496	20383	792
2–MER	1S	640109	441835	251589	29885	827
	2S	552408	441835	288254	20383	598
3–MER	1S	640109	441835	234604	11718	827
	2S	535056	441835	231877	11718	827
4–MER	1S	640111	441837	224868	11718	827
	2S	623308	441837	222141	11718	827
RDM	1S	534787	229202	229671	18746	792
	2S	494069	229202	238814	18746	792
HMM	1S	681664	459265	2745336	27514	792
BN	1S	1104860	458422	934156	40766	1196

Table IV.4: Gap length for length < 10 nucleotides

Model	Score	ETS1	YY1	SPI1	USF1	E2F1
PWM	1S	146	226	18	4906	126262
	2S	146	226	21	4906	126262
2–MER	1S	156	226	397	3346	120918
	2S	181	226	346	4906	167224
3–MER	1S	156	226	426	8533	120918
	2S	186	226	431	8533	120918
4–MER	1S	156	226	444	8533	120918
	2S	160	226	450	8533	120918
RDM	1S	186	436	435	5334	126262
	2S	202	436	418	5334	126262
HMM	1S	146	217	36	3634	126262
BN	1S	90	218	107	2453	83612

Table IV.3 and Table IV.5 are samples of our specificity results for TFBSs' of length less than and greater than (or equal to) ten nucleotides respectively. Specificity is represented

as number of hits (putative TFBSs with score above threshold).We count the hits over our random background. In general the number of hits increases with the increase of TFBS lengths. For similar lengths, RDM performs better (have less number of hits) on several occasions than other models.

Table IV.4 and Table IV.6 are samples of our other specificity results for TFBSs of length less than and greater than (or equal to) ten nucleotides respectively. Gap length is defined as *Background–Length/Number–of–Hits*. Since it is inversely proportional to the total number of hits, these tables have opposite trends with respect to Table IV.3 and Table IV.5.

Table IV.5: Number of hits for length ≥ 10 nucleotides

Model	Score	MAX	MEF2A	NFKB1	SRF	STAT1
PWM	1S	8160	19902	3666	784	3457
	2S	6524	17492	2034	543	4641
2–MER	1S	2304	26175	1224	712	686
	2S	2304	13904	964	905	644
3–MER	1S	1774	7890	823	778	475
	2S	1774	6856	763	575	402
4–MER	1S	1774	12349	401	987	368
	2S	1841	12349	401	987	380
RDM	1S	775	7152	284	167	47
	2S	775	7326	284	167	47
HMM	1S	12378	16546	3817	968	1849
BN	1S	4270	47616	6190	536	1622

IV.3 Ranking Analysis

To evaluate the performance of different models we introduce a ranking system among them. The basic criteria of good models are that they predict correct TFBSs as much as possible (higher sensitivity) and give false positives as low as possible (higher specificity). To increase sensitivity we need to lower the cut–off threshold so that more and more true positives can be detected. To increase specificity we need to increase cut–off threshold so

Table IV.6: Gap length for length ≥ 10 nucleotides

Model	Score	MAX	MEF2A	NFKB1	SRF	STAT1
PWM	1S	12254	5024	27277	127551	28926
	2S	15328	5716	49164	184162	21547
2–MER	1S	43402	3820	81699	140449	145772
	2S	43402	7192	103734	110497	155279
3–MER	1S	56369	12674	121506	128534	210526
	2S	56369	14585	131061	173913	248756
4–MER	1S	56369	8097	249376	101317	271739
	2S	54318	8097	249376	101317	263157
RDM	1S	129032	13982	352112	598802	2127659
	2S	129032	13650	352112	598802	2127659
HMM	1S	8078	6043	26198	103305	54083
BN	1S	23419	2100	16155	186567	61652

that more and more false positives are discarded. Due to this contradictory relation with threshold, one has to select a certain sensitivity to compare the performance with respect to specificity. This analysis can be done using ROC curve. Since we want to settle for a sensitivity as high as $\geq 90\%$, we calculate the specificity of the models at that level. We rank the models according to their specificity. Since we compare a total of 12 models, the rank position of each model can be anything between 1 and 12. In case of a tie, we prefer a simpler model.

Table IV.7 and Table IV.8 show our ranking results for TFBSs of length less than and greater than (or equal to) ten nucleotides respectively. RDM perform better (have top ranks) on several occasions and more so for longer motifs.

IV.4 Model Complexity Analysis

Being a very specific model might not always suffice. A very complicated model with a lot of parameters can over–fit the training data resulting in very high sensitivity and specificity. But that model may not generalize well for new test data. So, we always strive to find a simple model with as few parameters as possible to fit the training data. We analyze our

Table IV.7: Ranking for length < 10 nucleotides

Model	Score	ETS1	YY1	SPI1	USF1	E2F1
PWM	1S	10	3	12	7	2
	2S	10	3	11	7	2
2–MER	1S	6	3	7	11	7
	2S	4	3	8	7	1
3–MER	1S	6	3	5	1	7
	2S	3	3	4	1	7
4–MER	1S	8	9	2	1	7
	2S	5	9	1	1	7
RDM	1S	2	1	3	5	2
	2S	1	1	6	5	2
HMM	1S	9	12	10	10	2
BN	1S	12	11	9	12	12

Table IV.8: Ranking for length ≥ 10 nucleotides

Model	Score	MAX	MEF2A	NFKB1	SRF	STAT1
PWM	1S	11	10	10	8	11
	2S	10	9	9	4	12
2–MER	1S	7	11	8	6	8
	2S	7	7	7	9	7
3–MER	1S	3	4	6	7	6
	2S	3	1	5	5	5
4–MER	1S	3	5	3	11	3
	2S	6	5	3	11	4
RDM	1S	1	2	1	1	1
	2S	1	3	1	1	1
HMM	1S	12	8	11	10	10
BN	1S	9	12	12	3	9

models to see how efficient they are. We define efficiency as gap length per parameter. At certain sensitivity, we want as large gap length as possible with as few parameters as can be used. Number of parameters of the models are calculated from the number of independent parameters those are necessary to define the model adequately.

For PWM models the total number of parameters is $3 \times N$ where N is the number of positions. For each position there are 4 different nucleotide choices and the distribution

Table IV.9: Gap length/parameter(maximum) for length < 10 nucleotides

Model	Score	ETS1	YY1	SPI1	USF1	E2F1
PWM	1S	8.11	12.56	0.86	233.62	5260.92
	2S	7.68	11.89	0.95	223	5050.48
2–MER	1S	2.08	3.01	4.41	37.18	1151.6
	2S	2.38	2.97	3.8	53.91	1577.58
3–MER	1S	0.62	0.9	1.35	27.09	319.89
	2S	0.74	0.89	1.36	27	319.04
4–MER	1S	0.2	0.3	0.44	8.37	94.84
	2S	0.21	0.3	0.44	8.36	94.76
RDM	1S	0.83	1.94	1.38	16.93	300.62
	2S	0.89	1.93	1.32	16.88	299.91
HMM	1S	6.08	9.04	1.29	129.79	3945.69
BN	1S	0.24	0.57	0.06	5.52	674.29

contains 3 independent parameters. For the MCM of order k the number of total parameters is $(N-k) \times (4^{k+1}-1)$, since there are total $N-k$ distributions and each distribution has $(4^{k+1}-1)$ free parameters. The RDM has total $15 \times N \times (N-1)/2$ parameters because there are total $N \times (N-1)/2$ possible unique pairs and the distribution of each pair contains 15 free parameters. The two score version of all the above three mentioned models have one extra parameter which is the threshold to determine criticality. For HMM we assume a first order profile hidden HMM with N matching states and $N/2$ insert states not occurring in the beginning or end. So, the emission matrix has $3 \times N$ parameters and the state transition matrix has $N/2$ free parameters for entering into insert states and $N/2$ free parameters for exiting the insert states. So, total number of parameters is $4 \times N$. For BNM, we use the result from [11]. According to [11], in an optimal Bayesian network learned using the minimum description length [19] scoring function where all the random variables (here positions) draw values from same set of k distinct values (here $k = 4$), we do not have to consider parent set with cardinality larger than $\log_k n$, where n is the total number of instances. If we assume that in our case each of the variable has this number of parents then the total number of parameters for Bayesian network is $N \times 4^{\log_4 n+1} - 4$.

Table IV.10: Gap length/parameter(maximum) for length ≥ 10 nucleotides

Model	Score	MAX	MEF2A	NFKB1	SRF	STAT1
PWM	1S	408.47	167.47	826.58	3543.08	688.71
	2S	494.45	184.39	1446	4977.35	501.09
2–MER	1S	321.5	28.3	544.66	851.21	747.55
	2S	319.13	52.88	686.98	665.64	792.24
3–MER	1S	111.84	25.15	214.3	204.02	278.47
	2S	111.62	28.88	230.74	275.61	328.61
4–MER	1S	31.58	4.54	122.24	44.15	96.88
	2S	30.41	4.53	122.18	44.13	93.78
RDM	1S	191.16	20.71	426.8	604.85	1558.72
	2S	190.88	20.19	426.29	604.24	1557.58
HMM	1S	201.95	151.08	595.41	2152.19	965.77
BN	1S	36.82	3.30	23.08	244.20	280.24

Table IV.9 and Table IV.10 are samples of our parameter efficiency results for TFBSs of length less than and greater than (or equal to) ten nucleotides respectively. It can be noted that although the number of parameters increases with the length of TFBSs, they give higher efficiency (bigger gap per parameters). However, there is a limit to the scenario when we are finding hits over the background. Without finding any hit in case of extremely restrictive models, we can not have meaningful efficiency statistics.

IV.5 Summary and Discussion

In Table IV.11, we accumulated some aggregated statistics about our models. We calculate the statistics from the results of 587 alignments and 12 models. The first column contains the model name and second column tells the scoring scheme (scalar scoring versus two–score vector scoring). The gap length, sensitivity, ranking, parameter and efficiency are those quantities defined in the previous sections. But in this table all the values are actually an average quantity over the 587 models. Finally the % of best among 587 alignments represents the percentage of alignments for which the corresponding model is selected as the best according to the specificity criteria defined in the previous sections.

Table IV.11: Aggregated measures for all models

Model	Score	Gap length	Sensitivity	Rank	Parameter	Efficiency	% of Best
PWM	1S	61971.87	94.26	10.03	31.86	1699.19	2.091
	2S	59303.48	94.10	10.09	32.86	1590.32	0.000
2–MER	1S	119421.30	94.47	7.10	144.30	666.32	1.141
	2S	119245.44	94.29	6.95	145.30	657.55	0.951
3–MER	1S	261864.72	95.08	5.01	543.05	356.85	1.331
	2S	258271.64	95.10	4.80	544.05	352.06	1.521
4–MER	1S	559146.17	95.93	3.35	1943.04	193.71	11.597
	2S	565621.39	96.19	3.29	1944.04	195.61	10.076
RDM	1S	1091423.40	95.68	2.21	831.25	783.04	44.677
	2S	1091819.99	95.48	1.78	832.25	782.91	23.954
HMM	1S	63055.05	93.88	9.23	42.48	1295.69	2.471
BN	1S	63297.88	93.72	9.66	616.08	257.77	0.190

Table IV.11 gives us a summary about the models. The gap length, ranking and percentage when the best specificity is obtained give a measure of how specific a model is. RDM performs the best with these criteria with higher order Markov chains behind it. PWM is the worst in terms of specificity with HMM and BN a little better. In parameter count and efficiency the scenario is almost reverse. The best performer is PWM, which explains why the model is still widely in use despite having a terrible specificity. PWM is not only the most concise model with the fewest number of parameters but also achieves the highest specificity per parameter. The next best seems to be HMM and surprisingly RDM. This reveals the efficacy of pairwise RDMs. These models are not too inefficient in terms of using parameters and can achieve remarkably high level of specificity. This analysis can provide a guideline for researchers looking for appropriate models for TFBSs in specific types of applications.

Chapter V

CONCLUSIONS

In this study, we compared different probabilistic models for TFBSs and developed relevant statistics to evaluate the model performance. We also introduce introduce for some models previously having only scalar scoring a vector scoring of matches of a model to the DNA sequence. We built a pipeline for motif model assessment. In the next stage we will build a database of motif models from our generated data. We also plan to do the similar comparison over ChIP-Seq data, which are experimentally more reliable.

REFERENCES

[1] Boyle AP, Guinney J, Crawford GE, and Furey TS. F-Seq: a feature density estimator for high-throughput sequence tags. *Bioinformatics (Oxford, England)*, 24(21):2537–8, November 2008.

[2] Ren B, Robert F, Wyrick JJ, Aparicio O, Jennings EG, Simon I, Zeitlinger J, Schreiber J, Hannett N, Kanin E, Volkert TL, Wilson CJ, Bell SP, and Young RA. Genome-wide location and function of dna binding proteins. *Science*, 290(5500):2306–2309, 2000.

[3] V. B. Bajic, A. Chong, S. H. Seah, and V. Brusic. An intelligent system for vertebrate promoter recognition. *IEEE Intelligent Systems*, 17:64–70, 2002.

[4] V. B. Bajic, S. H. Seah, A. Chong, S. P. T. Krishnan, J. L. Y. Koh, and V. Brusic. Computer model for recognition of functional transcription start sites in rna polymerase ii promoters of vertebrates. *Journal of Molecular Graphics and Modelling*, 21(5):323–332, 2003.

[5] Y. Barash, G. Elidan, N. Friedman, and T. Kaplan. Modeling dependencies in protein-dna binding sites. In *Proceedings of the Annual International Conference on Computational Molecular Biology, RECOMB*, pages 28–37, 2003.

[6] I. Ben-Gal, A. Shani, A. Gohr, J. Grau, S. Arviv, A. Shmilovici, S. Posch, and I. Grosse. Identification of transcription factor binding sites with variable-order bayesian networks. *Bioinformatics*, 21(11):2657–2666, 2005.

[7] P. V. Benos, M. L. Bulyk, and G. D. Stormo. Additivity in protein-dna interactions: How good an approximation is it? *Nucleic Acids Research*, 30(20):4442–4451, 2002.

[8] M. L. Bulyk, P. L. F. Johnson, and G. M. Church. Nucleotides of transcription factor binding sites exert interdependent effects on the binding affinities of transcription factors. *Nucleic Acids Research*, 30(5):1255–1261, 2002.

[9] Tuerk C and Gold L. Systematic evolution of ligands by exponential enrichment: Rna ligands to bacteriophage t4 dna polymerase. *Science*, 249(4968):505–510, 1990.

[10] F. Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[11] N. Dojer. Learning bayesian networks does not have to be np-hard. 4162 LNCS:305–314, 2006.

[12] Latchman DS. Transcription factors: An overview. *The International Journal of Biochemistry and Cell Biology*, 29(12):1305 – 1312, 1997.

[13] R. D. Finn, J. Clements, and S. R. Eddy. Hmmer web server: Interactive sequence similarity searching. *Nucleic Acids Research*, 39(SUPPL. 2):W29–W37, 2011.

[14] G. Z. Hertz and G. D. Stormo. Identifying dna and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, 15(7-8):563–577, 1999.

[15] Watson JD. *Molecular Biology of the Gene*. Pearson, San Fransisco, CA, sixth edition edition, 2008.

[16] Ho JW, Bishop E, Karchenko PV, Ngre N, White KP, and Park PJ. ChIP-chip versus ChIP-seq: Lessons for experimental design and data analysis. *BMC Genomics*, 12(1):134, February 2011.

[17] Connaghan-Jones KD, Moody AD, and Bain DL. Quantitative DNase footprint titration: a tool for analyzing the energetics of protein-DNA interactions. *Nature Protocols*, 3(5):900–14, January 2008.

[18] K. J. Kechris, E. van Zwet, P. J. Bickel, and M. B. Eisen. Detecting dna regulatory motifs by incorporating positional trends in information content. *Genome biology*, 5(7), 2004.

[19] Wai Lam and Fahiem Bacchus. Learning bayesian belief networks: An approach based on the mdl principle. *Computational Intelligence*, 10(3):269–293, 1994.

[20] L. P. Lim and C. B. Burge. A computational analysis of sequence features involved in recognition of short introns. *Proceedings of the National Academy of Sciences of the United States of America*, 98(20):11193–11198, 2001.

[21] Das MK and Dai HK. A survey of dna motif finding algorithms. *BMC Bioinformatics*, 8(Suppl 7):S21, 2007.

[22] Park PJ. ChIP-seq: advantages and challenges of a maturing technology. *Nature Reviews. Genetics*, 10(10):669–80, October 2009.

[23] Lawrence R. Rabiner. Tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[24] Brooke Rhead, Donna Karolchik, Robert M. Kuhn, Angie S. Hinrichs, Ann S. Zweig, Pauline A. Fujita, Mark Diekhans, Kayla E. Smith, Kate R. Rosenbloom, Brian J. Raney, Andy Pohl, Michael Pheasant, Laurence R. Meyer, Katrina Learned, Fan Hsu, Jennifer Hillman-Jackson, Rachel A. Harte, Belinda Giardine, Timothy R. Dreszer, Hiram Clawson, Galt P. Barber, David Haussler, and W. James Kent. The UCSC Genome Browser database: update 2010. *Nucleic Acids Research*, 38(Database issue):D613–619, 2010.

[25] A. Sandelin, W. Alkema, P. Engstrm, W. W. Wasserman, and B. Lenhard. Jaspar: An open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Research*, 32(DATABASE ISS.):D91–D94, 2004.

[26] Geir Kjetil Sandve and F Drabløs. Drabløs. a survey of motif discovery methods in an integrated framework. *Biology Direct*, 1(11), 2006.

[27] R. Siddharthan. Dinucleotide weight matrices for predicting transcription factor binding sites: generalizing the position weight matrix. *PloS One*, 5(3), 2010.

[28] G. D. Stormo. Dna binding sites: Representation and discovery. *Bioinformatics*, 16(1):16–23, 2000.

[29] G. D. Stormo, T. D. Schneider, L. Gold, and A. Ehrenfeucht. Use of the 'perceptron' algorithm to distinguish translational initiation sites in e. coli. *Nucleic Acids Research*, 10(9):2997–3011, 1982.

[30] T. Werner. Models for prediction and recognition of eukaryotic promoters. *Mammalian Genome*, 10(2):168–175, 1999.

[31] E. Wingender, P. Dietze, H. Karas, and R. Knppel. Transfac: A database on transcription factors and their dna binding sites. *Nucleic Acids Research*, 24(1):238–241, 1996.

[32] Eric P. Xing, Michael I. Jordan, Richard M. Karp, and Stuart Russell. A hierarchical bayesian markovian model for motifs in biopolymer sequences. In *In Proc. of Advances in Neural Information Processing Systems*, pages 200–3. MIT Press, 2003.

[33] Qi Y, Rolfe A, MacIsaac KD, Gerber GK, Pokholok D, Zeitlinger J, Danford T, Dowell RD, Fraenkel E, Jaakkola TS, Young RA, and Gifford DK. High-resolution computational models of genome binding events. *Nature Biotechnology*, 24(8):963–70, August 2006.

[34] X. Zhao, H. Huang, and T. P. Speed. Finding short dna motifs using permuted markov models. *Journal of Computational Biology*, 12(6):894–906, 2005.