Cross-Context Prediction of Transcription Factor Binding Sites for Unannotated ENCODE Data

A Thesis

Presented to

the Faculty of the School of Engineering and Applied Science

University of Virginia

In Partial Fulfillment

of the requirements for the Degree

Master of Science (Computer Science)

by

Ritambhara Singh

December 2014

Abstract

Finding transcription factor (TF) binding sites in the DNA is of central importance for understanding the molecular mechanisms of gene regulation. The genome-wide identification of TF-binding sites (TFBS) has recently become available for a few species owing to the ENCODE consortium through ChIP-seq. However, due to the expensive and time-consuming nature of existing ChIP-seq technologies, computational methods, for accurately modeling the DNA sequence preference of transcription factors and for predicting their binding sites, continue to be critical, especially for the under-studied cellular contexts, e.g. non-model species' genomes or rare disease's cell types. Most existing studies for sequence-driven TF binding prediction have ignored the consideration of cellular contexts and/or ignored the data heterogeneity among multiple contexts. We, on the other hand, propose a novel method named "transfer string kernel" (TSK) to achieve better predictions of sequence-based TF binding preferences using cross-context sample adaptation. Relying on the idea of "knowledge transfer" from machine-learning, TSK pursues context-specific TF binding prediction via three essential components: (1) TF binding patterns on DNA sequences are mapped to a high-dimensional feature space under the discriminative string kernel framework; (2) Labeled TF binding examples from a source context are transferred to a target context through re-weighting the source samples adaptively using kernel mean matching (KMM) estimator; (3) An instance-weighted support vector machine framework is then implemented to classify sequence segments into TF binding or non-binding sites in the target context. Utilizing TF binding data from ENCODE, we experimentally verify TSK's capability to adapt from the source cell-type GM12878 to the target cell-type K562 and the source genome as human to the target genome as mouse. The proposed TSK method consistently improves the predictions of TF binding in these target contexts over the baselines, without consideration of heterogeneity or scarcity among samples, and state-of-the-art sequence-based TF binding predictors.

Approval Sheet

This thesis is submitted in partial fulfillment of the requirements for the degree of

Master of Science (Computer Science)

Ritambhara Singh

This thesis has been read and approved by the Examining Committee:

Gabriel Robins, Advisor

Yanjun Qi, Advisor

James Cohoon, Committee Chair

Worthy Martin

Mazhar Adli

Accepted for the School of Engineering and Applied Science:

James H. Aylor, Dean, School of Engineering and Applied Science

December 2014

To my parents and my sister, for their undying support and faith.

Acknowledgments

I would like to thank all my advisors: Professors Gabriel Robins, Yanjun Qi and Mazhar Adli. Professor Robins for helping me begin my journey into the field of computational biology and his constant guidance at every step. Professor Qi for introducing me to the world of Machine Learning, with its vast potential, and helping me find a project when I had run out of ideas. She has worked with me like a colleague and has been a wonderful mentor from the very first day. Professor Adli for giving me the opportunity to work in his lab as a computational biologist and being extremely patient and helpful, when I struggled to get accustomed to the concepts of biology and bioinformatics tools. Everything I have achieved is a result of insightful discussions, help and support from all of them.

I am extremely grateful to Dr. Ryan Layer, who ensured my smooth transition into the field of bioinformatics and has always been available to address concerns and discuss new ideas. Thank you to Professor Aaron Quinlan, for allowing me to attend his lab meetings and helping me find a lab to work in, during my first year. I am also grateful to Royden Clark, a good teacher and a friend, to whom I owe my UNIX scripting skills.

I would like to thank all members (current and former) of Adli Lab: Jeremy Thorpe, Ali Osman Bayrak, Dr. Cem Cusku, Dr. Mahmut Parlak, Dr. Sevki Arslan and Stephen Shang, for directing me towards important problems in biology and providing encouragement when I got stuck. I would also like to thank my colleagues from Machine Learning and Bioinformatics Lab: Tung Dao, Beilun Wang, Weilin Xu, Sarah Masud Preum and Nicholas Janus, for talks and discussions that have constantly improved my understanding of Machine Learning.

Last, but not the least, heartfelt gratitude to my family members and friends, near and far, for being a strong network of support and encouragement during difficult times.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The activity of genes, in a living cell, is coordinated by a regulatory network in which transcription factor (TF) proteins are essential. Each TF protein binds to specific binding sites in the regulatory regions (e.g. promoters, enhancers) of the genes, thus providing their activation or repression. Accurate modeling of TF binding sites (TFBS) on DNA is key for understanding the function and evolution of genomes. Only recently, has the genome-wide identification of TF-binding sites become possible, thanks to the development of chromatin immunoprecipitation followed by massively parallel DNA sequencing (ChIP-seq) technologies [2]. However, maps of genome-wide binding sites are only available for a few of the cell types and that too only in the human and mouse genomes. Major reason for this gap is that performing ChIP-seq experiment on the element of interest is both time and money consuming. Hence, computational methods to identify the binding sites of DNA-associated proteins are constantly the key focus, which aim to predict the relative affinity (or dissociation constant) and/or the probability of occupancy at any position in the genome.

Previous literature has included a large number of sequence-motif based computational techniques, which normally use position-based sequence information for predicting TFBS. For example, [3] applies greedy optimization with bootstrapping, using coverage profiles as motif positional preferences, for better identification of transcription factor motifs than existing tools like [4].

Another category of models are string kernels, that are trained by extracting features from a set of previously labeled sequences and then use the learned models to classify new set of sequences. Very recently, [1] extended this discriminative classification setup for predicting sites with transcription factor binding in a cell-type specific manner, by using the epigenetic information (e.g ChIP-seq maps and histone modifications) of another cell type. The authors have implemented string kernels under support vector machines (SVM) framework, in order to label a DNA sequence position as positive or negative for binding of a certain transcription factor. Their results have indicated that the SVM string kernel based models achieve better binding site predictions than traditional motif based approaches.

Despite the exciting aforementioned work of string kernel based TFBS prediction, one gap is significant: this framework assumes the training and testing samples are drawn from the same probability distribution. However, this assumption can be easily violated for the TFBS prediction tasks due to two reasons: (a) the labeled data is scarce. ENCODE datasets have covered only a limited number of cell types [5] and have only involved human and mouse genomes. (b) the labeled datasets are heterogeneous, since the DNA sequence pattern from TFBS varies with many factors, e.g. certain TF, certain cell type, certain genome or certain environmental condition. As an attempt to solve these two challenges, we propose a novel approach named "Transfer String Kernel (TSK)" to achieve better predictions for TF binding preferences through knowledge transferring across the samples. To demonstrate the effectiveness of TSK, we apply it on two very different TFBS applications, wherein one is for cross cell-types, and the other is for cross-genomic TFBS predictions. Our method exhibits the following advantages,

- A major advantage is the natural transfer of knowledge/samples among heterogeneous sample groups, e.g. from one genome, cell type or environmental condition, to another. This is extremely important when limited numbers of samples are available in the target group and there exists heterogeneity among the groups.

- Secondly, the discriminative string kernel framework with mismatches provides a flexible and principled framework in capturing both the motif mismatch patterns and at the same time considering global distribution among the samples.

Chapter 2

Background

2.1 Target Biological Task

2.1.1 Transcription Factors

Transcription Factors (TFs) are proteins that bind to sequence-specific locations on the DNA and control the rate of transcription of genetic information from DNA to messenger RNA. They are essential for the regulation of gene expression in the cell and play a vital role in many cellular processes. Genes are often flanked by several Transcription Factor binding sites (TFBS) for distinct TFs, and efficient expression of each of these genes requires the cooperative action of several different TFs. Hence, the combinatorial use of a subset of approximately 2000 human transcription factors easily accounts for the unique regulation of each gene in the genome during cell development. Binding event of a TF provides deeper understanding of cell machinery. Many disorders/diseases, which are caused by up or down regulation of genes can be explained by such binding or non-binding events. Thus, finding all TFBS on the genome becomes an important problem in the field of biology. Recently, Chromatin immuno-precipitation followed by massively parallel DNA sequencing (ChIP-seq) technologies [2], have allowed genome-wide identification of TFBS (Figure 2.1). However, performing ChIP-seq experiments on all possible TFs in hundreds of cell types and genomes is a time and money consuming process. The largest repository of TFBS data obtained through ChIP-seq, ENCODE [6], has sampled 119 of 1800 known TFs for a limited number of cell types in human and mouse genomes [5]. Therefore, due to this gap in available information regarding TFBS, a large number of tools have been developed to predict the binding sites of TFs in a particular cell type or genome.

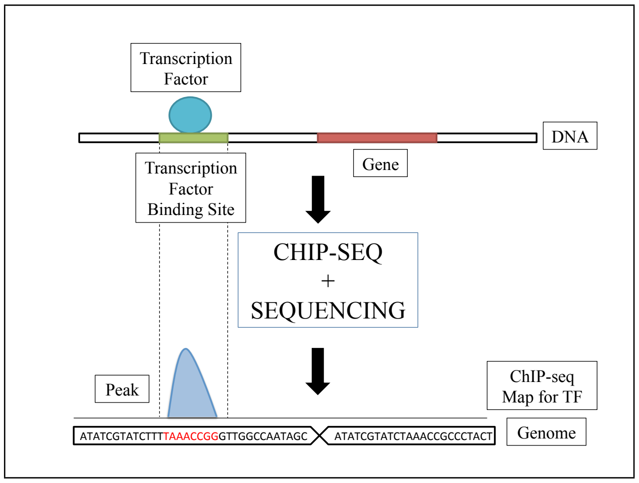


Figure 2.1: ChIP-seq technology allows genome-wide mapping of TFBS. After sequence alignment step, ChIP-seq maps consist of “peaks” or enriched regions marking binding locations of TFs.

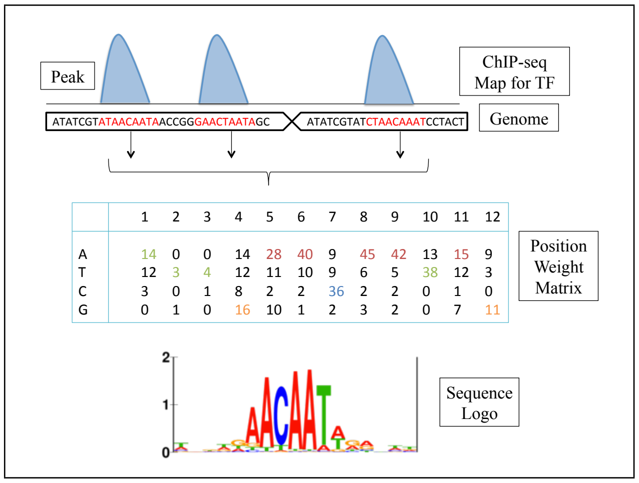
2.2 State-of-the-art Bioinformatics Tools

2.2.1 Position Weighted Matrix based Approaches

Earlier developed techniques are generative in nature, that is based on input sequences, a Position Weight Matrix (PWM) is computed. PWM is a probabilistic or weighted description of sequences and is used to describe consensus sequence motifs of TFBS. It is generally represented in the form of a *sequence logo* as illustrated in figure 2.2. The computed PWM is then compared to existing PWM databases like JASPAR [7], and the tool outputs a list of TFs ranked according to the probability of their binding to the given set of sequences. Tools using PWM calculation for TFBS prediction, like MEME [4], are very popular. However, recent DREAM competition [8] has compared a large number of existing models that predict potential transcription factor binding preferences based on sequences. It concluded that those simple models based on position weight matrices (PWM) , even when trained using best methods, perform similarly for majority of transcription factors being examined.

2.2.2 String Kernel based Approaches

On the other end of the spectrum, are discriminative techniques, like string kernels, that have been used successfully for capturing patterns of DNA or Protein sequences through local string comparisons with mismatches under the discriminative classification framework. This discriminative setting tries to capture the differences *among* classes, while its counterpart, generative models like tools from [4] focus on capturing shared characteristics *within* classes.Previous studies [1] have shown that the discriminative models have better distinction power over the motif based generative models. The idea of using string kernels for classifying



	1	2	3	4	5	6	7	8	9	10	11	12
A	14	0	0	14	28	40	9	45	42	13	15	9
T	12	3	4	12	11	10	9	6	5	38	12	3
C	3	0	1	8	2	2	36	2	2	0	1	0
G	0	1	0	16	10	1	2	3	2	0	7	11

Figure 2.2: Sequences under TFBS are used to calculate Position Weight Matrix, describing the TF motif. A sequence logo graphically represents the sequence conservation of nucleotides in a set of DNA sequences.

biological sequences was first introduced in [9] [10], where they were used to classify protein sequences into different protein families. Later, due to TFBS data availability through ChIP-seq, string kernels were used to classify sequences of DNA into TF binding and non-binding sites in [1]. The authors implemented di-mismatch string kernel to convert input sequences into features, which were then used to train Support Vector Machine for binary classification. The model was trained on sequences from one cell type and used to make prediction of binding events on sequences from another cell type. The paper compared the performance of this model with various motif-finding tools (like MDscan, cERMIT etc), evaluated on the ability to discriminate between ChIP-Seq defined peaks and nearby non-peak regions. A ChIP-seq peak is a region of significant enrichment, indicating evidence of TF binding event. The SVM sequence model was shown to better capture the underlying sequence content of TF binding sites for > 90 % of TFs. However, most of the string kernel implementations use “Trie” data structure for kernel computation, which results in slow run times for large values of mismatches in the sequences. To solve this problem, [11] introduced “Intersection based algorithm” for more efficient and scaleable implementation of string kernels, thus improving over the running times of previously discussed methods.

Building on the idea of cross-cell type training and prediction of TFBS sequences, we expand into the notion of “cross-context” prediction by taking into consideration data heterogeneity and scarcity. We also implement the efficient versions of string kernel, resulting in faster computations and add transfer learning in order to enable the model to make predictions not only across different cell types but also across genomes. Our model, “Transfer String Kernel”, has been explained and evaluated in subsequent chapters.

Chapter 3

Transfer String Kernel

In this work, we focus on string kernel-based classification setup, which aims to classify each DNA sequence segment as TFBS or non-binding background site. Currently available TFBS datasets exhibit both properties of scarcity and heterogeneity. Thus, existing models are not sufficient for capturing differences among samples or data distributions, especially when working with TFBS of separate genomes, cell-types, environments etc.

To improve the cross-context sequence-based predictions of TFBS, we propose a novel learning approach, "Transfer String Kernel" (TSK) here, in order to enable discriminative TFBS predictions by utilizing available TF binding labels in an efficient way. We now discuss the different components of our approach.

3.1 Support Vector Machines

Support Vector Machines (SVMs) are a class of supervised learning algorithm. The goal of SVM is to learn a linear decision boundary to discriminate between the class of a set of positive and negative examples of the input training vectors. Once trained, we get a linear classifier, which can be used to classify test samples. Thus given training data,

$$x_1, x_2, \ldots, x_n \in R_n$$
$$y_1, y_2, \ldots, y_n \in \{-1, +1\}$$

the linear classification rule f can be specified by a pair $\langle w, b \rangle$ where $w \in R_n$ and $b \in R$, such that,

$$f(x) = \langle w, x \rangle + b \tag{3.1}$$

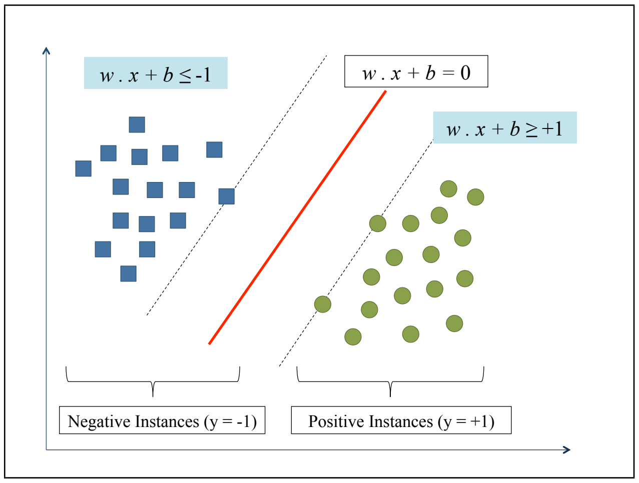


Figure 3.1: Linear SVM Classifier.

where a point x is classified as positive, if $f(x) > 0$, else if $f(x) < 0$, it is classified as a negative example as seen in Figure 3.1.

The simplest approach is that SVMs find the hyper-plane that maximizes the gap between data points (positive from negative) on the boundaries (so-called "support vectors") [12]. This is known as the "hard margin"; however, this optimization does not work well in case of non-linearly separable data. In that case we use the concept of "soft margin", where a 'slack variable' is assigned to each instance $\xi \geq 0$, which can be thought of distance from the separating hyper-plane if an instance is misclassified and 0 otherwise. Another solution for this optimization problem is to project the input data into a feature map, such that it can be classified linearly by the SVM and thus kernels are used for this type of implementation.

3.2 Kernels

Kernels are used to map the input data into feature space, thus allowing SVM to linearly classify the feature space for non-linearly separable data, as presented in Figure 3.2.

The smooth combination of kernel techniques with SVMs is a result of some interesting properties of SVMs. The important feature is that SVM optimization problem is equivalent to solving a dual quadratic problem [9]. Quadratic Problem (QP) is a special optimization problem where the function to optimize ("objective") is quadratic, subject to linear constraints. Thus the SVM optimization problem, like QP, can be fitted in the "dual form" with weights $\alpha_i, i = 1 \ldots N$:

$$Maximize \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i \vec{x}_j \tag{3.2}$$

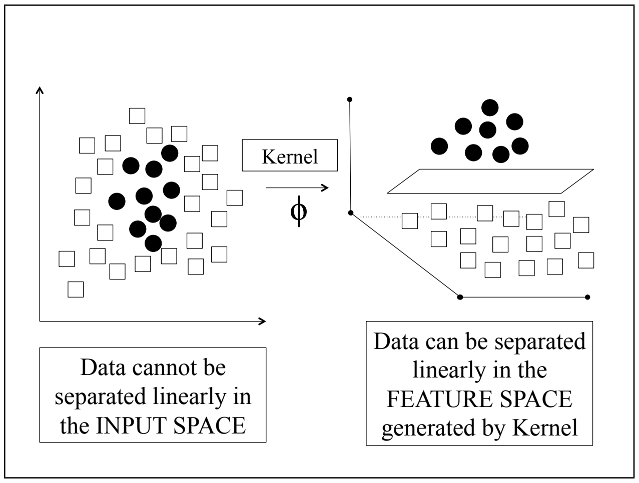

Kernel
ϕ
Data cannot be separated linearly in the INPUT SPACE
Data can be separated linearly in the FEATURE SPACE generated by Kernel

Figure 3.2: Example of Kernel mapping of input data into feature space.

subject to

$$\alpha_i \geq 0,$$
$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

This dual problem only depends on the inner products $\langle x_i, x_j \rangle$ and thus allows the combination of kernel techniques into the SVMs [12]. Therefore, if we have a set of labeled examples (x_i, y_i), where x_i belongs to input space χ, then any feature map ϕ from χ into a (higher-dimensional) vector is called feature space:

$$\phi : \chi \to R_n, \tag{3.3}$$

a kernel K on $\chi \times \chi$ can be obtained such that,

$$K(x, y) = \langle \chi(x), \chi(y) \rangle \tag{3.4}$$

And thus, by replacing $\langle x_i, x_j \rangle$ by $K(x_i, x_j)$ in the dual problem, SVMs can be used in feature space [9].

3.3 String Kernels.

Classic string kernel methods have provided some of the most accurate results in remote protein fold and homology detections [13],[11]. Recently, [1] proposed to utilize a similar k-mer based string kernel classification setup to predict cell-type specific TFBS, by using sequence information of these sites.

The key idea of string kernels is to apply a mapping $\phi(\cdot)$ to map sequences of arbitrary length into a vectorial feature space of fixed length. In this space, a standard classifier such as a support vector machine (SVM) ([14]) can be then applied. As SVMs require only inner products between examples in the feature space, rather than the feature vectors themselves, string kernel ([13],[11]) implicitly computes an inner product in this feature space:

$$K(X,Y) = \langle \phi(X), \phi(Y) \rangle, \tag{3.5}$$

where $X = (x_1, \ldots, x_{|X|})$, where $|X|$ means the length of the sequence. $X, Y \in S$, S is the set of all sequences composed of all possible nucleotides. $\phi : S \rightarrow R^m$ defines the mapping from sequence $X \in S$ to a m-dim. feature vector.

Our target problem of TFBS predictions essentially could be treated as the tasks of learning strings of nucleotides (ATCG) that typically describe the patterns of binding for a certain TF of interest. Given an input DNA sequence segment, the target binding relationship will be recognized by performing matching between patterns present in the test sequence segments and the patterns in the annotated training segment examples. This amounts to computing string kernel (similarity score) between DNA sequence segments based on certain feature representations.

Feature extraction and feature representation play key roles in the effectiveness of sequence analysis since biological sequences cannot be readily described as feature vectors. One classic representation is to treat DNA sequence segment as an unordered set of nucleotide k-mers (combinations of k adjacent nucleotide residues) and a feature vector indexed by all k-mers records the number of occurrences of each k-mer in the current sequence. The string kernel using this representation was named as *spectrum kernel* [13] , where the spectrum representation describes the counts of occurrence of each nucleotide k-mer in a DNA sequence segment. Similarity scores between sequences are then computed by taking an inner product between corresponding "k-mer - indexed" feature vectors

$$K(X,Y) = \sum_{\gamma \in \Gamma} c_X(\gamma) \cdot c_Y(\gamma), \tag{3.6}$$

where γ represents a k-mer, Γ_k is the set of all possible k-mers, and $c_X(\gamma)$ is the number of occurrences (with normalization) of k-mer γ in sequence X.

Efficient inner product evaluation becomes very important in case of k-mer based representations since using explicit feature vector mapping $\phi(\cdot)$ becomes problematic (especially in the case of inexact matching) for even small value k as the dimensionality of the feature vectors is exponential in k. Researchers have proposed various strategies ([13, 11]) to conquer these difficulties and we adopt a sufficient statistic strategy from [11] for efficient string kernel computation. We illustrate the concept of string kernels using (k, m)-mismatch

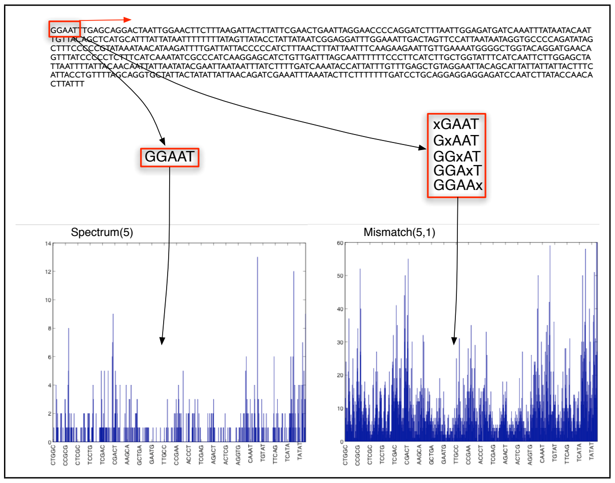


Figure 3.3: String kernel. A sliding window of size k=5 is used to derive feature representation $\phi(X)$ of an input sequence X.

kernel ([11]) in figure 3.3. Mismatch string kernel considers k-mer counts with m inexact matching of k-mers. As can be seen from Figure 3.3 string kernels with mismatches can capture binding patterns using combinations of nucleotide (k-mer with mismatch) as features.

3.4 Transfer Examples through Importance Re-weighting via Kernel Mean Matching.

In the above string kernel based discriminative framework, the training and testing samples are assumed to be drawn from the same probability distribution. However, as discussed previously, this assumption can be easily violated for the targeted TFBS prediction task due to two reasons:

- Labeled data is scarce. TFBS dataset is available for a limited number of cell types for human and mouse genomes only.
- Labeled datasets are heterogeneous. Since the DNA sequence pattern related to TF binding varies with many factors, e.g. certain cell type, certain genome or certain environmental condition.

Thus, it is important to figure out a certain strategy that could not only consider the variation among training sources and among training and testing samples, but also at the same time be able to combine as many training samples that are currently available as possible. We propose to revise the mismatch string kernel framework using “Transfer Learning” ([15]), which has been applied to a wide variety of real-world problems to perform knowledge transfer between source and target tasks. Knowledge transfer, if done successfully, would greatly improve the performance of learning by avoiding the much expensive data-labeling efforts [15].

Specifically we focus on revising the string kernel algorithms under the "covariate shift" assumption ([16]), i.e. the conditional probability distribution of the output variable given the input variable remains fixed in both the training and testing set. That is, the shift happens only for the marginal probability distribution of the covariates, not for the conditional distribution.

$$P_{tr}(dy|x) = P_{ts}(dy|x) = P(y|x) \tag{3.7}$$

This indicates that,

$$\begin{aligned} P_{tr}(x,y) &= P_{tr}(dy|x) * P_{tr}(x) = P(y|x) * P_{tr}(x) \\ P_{ts}(x,y) &= P_{ts}(dy|x) * P_{ts}(x) = P(y|x) * P_{ts}(x) \end{aligned}$$

Under the classical empirical risk minimization setting, we thus aim to estimate our string kernel classifier through the following optimization,

$$\begin{aligned} minR[P_{ts}, l(x,y,\theta)] = E_{P_{ts}}[l(x,y,\theta)] + \lambda\Omega[\theta] \\ = E_{P_{tr}}[\beta(x,y) * l(x,y,\theta)] + \lambda\Omega[\theta] \end{aligned}$$

It is well-known that under the setting of "covariate shift", the key to correct the sampling bias is to estimate the so-called "importance weight" or density ratio $\beta(x,y)$ for each training sample,

$$\begin{aligned} E_{P_{ts}}[l(x,y,\theta)] = E_{P_{tr}}[\beta(x,y) * l(x,y,\theta)] \\ = E_{P_{tr}}[\frac{P_{ts}(x,y)}{P_{tr}(x,y)} * l(x,y,\theta)] \end{aligned}$$

where when provided $P_{ts} \ll P_{tr}$, the importance weighting is defined as,

$$\beta_{imp}(x,y) := \frac{P_{ts}(x,y)}{P_{tr}(x,y)} = \frac{P_{ts}(x)}{P_{tr}(x)}, \tag{3.8}$$

A number of methods have been proposed to estimate the "importance weight" β from finite samples [17], including kernel mean matching (KMM), logistic regression, KL importance estimation and possibly many others. Here we focus on the KMM: kernel mean matching estimator from [16] to estimate β.

KMM estimator accounts for the difference between $P_{ts}(x, y)$ and $P_{tr}(x, y)$ by re-weighting the training points such that the means of the training and testing points in a Reproducing Kernel Hilbert space (RKHS) are close. That is why this re-weighting process is named as kernel mean matching (KMM). When the RKHS is universal, the population solution to this minimization is exactly the ratio $\frac{P_{ts}(x)}{P_{tr}(x)}$. Let β_i represents the "important weighting" of a source instance x_i. KMM uses a so-called "maximum mean discrepancy" measure to minimize the difference between the empirical mean of the source and the empirical mean of the target distribution. Formally, KMM tries to match the mean elements in a feature space induced by a kernel $k(\cdot, \cdot)$ on the domain $\mathcal{X} \times \mathcal{X}$,

$$min_{\beta}\hat{L}(\hat{\beta}) := \|\frac{1}{n_{tr}}\sum_{i=1}^{n_{tr}}\hat{\beta}_i * \phi(X_i^{tr}) - \frac{1}{n_{ts}}\sum_{i=1}^{n_{ts}}\phi(X_i^{ts})\|,$$

$$= \frac{1}{n_{tr}^2}\hat{\beta}^T K\hat{\beta} - \frac{2}{n_{tr}^2}\kappa^T\hat{\beta} + \text{const.}$$

Here, K is the kernel matrix among all source examples, and κ is the weighted sum of kernel values among source example and target examples,

$$\kappa_i = \frac{n_{tr}}{n_{ts}}\sum_{j=1}^{n_{ts}} K(x_i^{tr}, x_j^{ts}) \tag{3.9}$$

The above equation is a quadratic program which can be efficiently solved through the projected gradient optimization. The derived importance weights will then be used to re-weight training samples in the support vector machine classification algorithm, i.e., instance weighted SVM optimizes:

$$\sum_{p=1}^{n_{tr}}\alpha_p - 0.5\sum_{p,q}\alpha_p\alpha_q y_p y_q K(x_p^{tr}, x_q^{tr}),$$

$$s.t.\sum_p \alpha_p y_p = 0, \qquad s.t.\beta_p C \geq \alpha_p \geq 0$$

KMM has been successfully implemented for RKHS and Gaussian kernels. In this paper we apply it in conjugation with (k, m)-mismatch string kernel, thus enabling us to perform knowledge transfer across sequential data.

3.5 Transfer String Kernel:Methodology

Combining biological sample from multiple species has been shown to be helpful in a number of important tasks. For example, cross-species gene expression analysis has show to recover true associations between genes [18]. The basic motivation there is that many genes across species perform similar functions or share the same regulatory relations so that one can exploit information on related genes from multiple species. For our

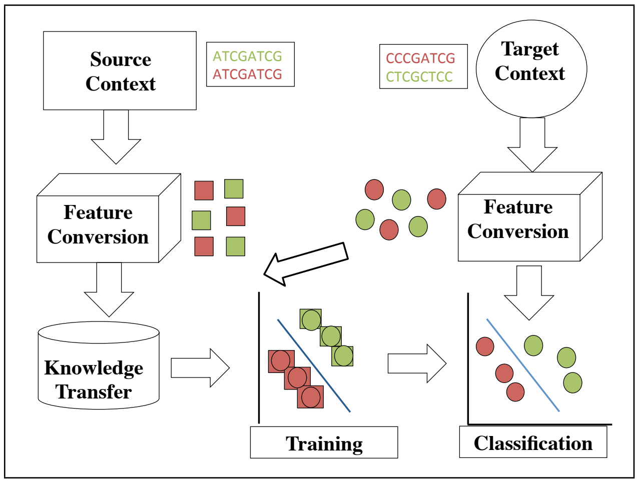


Figure 3.4: Summary of the Transfer String Kernel approach.

task, in addition to improving prediction quality, cross-species TFBS analysis can identify conserved/common regulatory relations, which are more likely to play essential roles, as well as species-specific regulatory relations. By combining the above subsections, the proposed "Transfer String Kernel" method takes a multi-stage style:

1. **String Kernel Step**: We first use mismatch string kernel to convert the DNA sequences into features and obtain the kernel matrix K and κ;
2. **Transfer Learning Step**: Next, we utilize Kernel Mean Matching on K and κ to obtain importance weights for each training sample $\hat{\beta}$;
3. **Instance Re-weighting and Classification Step**: Finally, we use $\hat{\beta}_i$ to re-weight our training samples and train an instance-weighted SVM classifier with the mismatch string kernel matrix, for performing classification of sequences using Support Vector Machines.

This approach has been summarized in Figure 3.4, and presents a number of notable properties:

- It is a wrapper approach and can be used to extend to all string kernel variations;
- It is very efficient as the string kernel has linear cost in the input length ;
- It provides a unified framework for TFBS prediction using training samples across many heterogeneous groups, e.g. cross-genomes;
- It is a general setup and not restricted to TFBS prediction task, since no task specific knowledge is necessary for the training;
- It can be used for other cross-context sequence analysis applications.

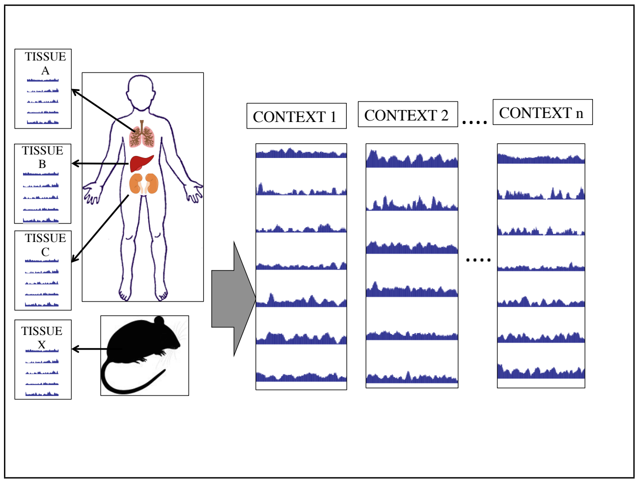


Figure 3.5: Examples of multiple cellular-contexts with urgent needs of TFBS predictions.

Chapter 4

Evaluation

4.1 Experimental Setup

To evaluate our approach, we apply TSK (Transfer String Kernel) on both cross-cell types and cross-genomic contexts. Our experimental setup is similar to that implemented in [1]. We used top 1000 TFBS sites obtained from the genome-wide mapping by ChIP-seq experiments as positive samples. Here coordinates of TFBS can be differentiated from background by significantly enriched sites, or "peak", and are obtained from the Chip-seq data available in ENCODE repository. However, our TFBS data differs from that of [1] because, instead of selecting flanking regions from the peaks as negative or non-binding sites, we selected the a "peak-centric standard" as described in [19]. "Peak centric standard" defines a single position in each ChIP-seq peak as a binding site for the TF (a positive), and treats all other genomic positions as negatives. Thus, 100 base-pairs long sequences from center of the peaks were labeled as positive TFBS sequences, while sequences labeled as negative or non-binding events (100 base-pairs in length) were chosen randomly from the genomic regions, which were not occupied by TF protein. Our aim is to generate datasets that imitate the scenario such that, given a set of sequences, our model is able to differentiate the TF binding regions from the non-binding background regions accurately. We first demonstrate the application of our model and its comparisons to other baselines on the task of cross-cell type TFBS learning and then on the cross-genome case. For cross-cell type case, we selected 11 common TFs among the two cell types GM12878 and K562 of human species. The former cell-type is related to blood cells, while the latter represents leukemia cells. Therefore, knowledge transfer of TFBS information is performed between healthy and diseased tissue types. For the cross-genome application, we selected 10 TFs for which, ChIP-seq data is available for cell types GM12878 and CH12 belonging to human and mouse species respectively. These two cell types are both relevant to blood cells and thus the 10 selected TFs, or regulatory proteins, help in the functioning of the cell machinery of blood cells in humans (GM12878 tissue) and mouse (CH12 tissue). The task of predicting TFs in one sample while utilizing data from a separate sample (across cell types or genomes) is useful in gaining knowledge of TFBS for a cell type for which ChIP-seq data is unavailable as well as provide deeper understanding of the relationship between TFBS of different cell types or genomes. All the datasets were equally divided into training and test sets and used as inputs for model evaluations.

4.2 Evaluation Metric

In this paper, our investigation of Transfer String Kernel is focused under the framework of SVM classifier. SVMs are known to provide state-of-the-art performance for many applications [14], in particular in computational biology [20]. Given sequence data of each sample, SVM learns a real-valued function that assigns a continuous score to each candidate sequence segment based on the labeled training set. Although -1, 1 prediction is usually obtained by simply taking the sign of this score, the value of the score itself contains some form of confidence in the prediction. Larger scores correspond to more confident predictions. As a result, we propose to rank all candidate segments in the test set by the value of the SVMs predictions. This naturally leads us to utilize AUC score from ROC curve as our main evaluation metric. Receiver Operator Characteristic (ROC) curves plot the true positive rate against the false positive rate for different cut-off values of the predicted score. The area under the ROC curve (AUC) is commonly used as a summary measure of diagnostic accuracy. AUC is interpreted as the probability that a randomly selected "event" will be regarded with greater suspicion (in terms of its continuous measurement) than a randomly selected "non-event". AUC score ranges between 0 and 1, where values close to 1 indicates more successful predictions.

The following first three sections present experimental results for the cross-cell-type data, while the last discusses the implementation of cross-genome TSK. Acronyms for different methods are listed as follows:

- **SK** represents the (k, m)-mismatch kernel implementation adapted from [11];
- **TSK** denotes our "Transfer String Kernel" approach, where transfer learning is performed once the kernel is computed using SK;
- **PWM** is the method of scoring each sequence for potential TF binding event based on Position Weight Matrices obtained from existing TF databases like JASPAR [7];
- **Di-mismatch Kernel** has been implemented in [1] for prediction across different cell types (but without considering the sample heterogeneity);

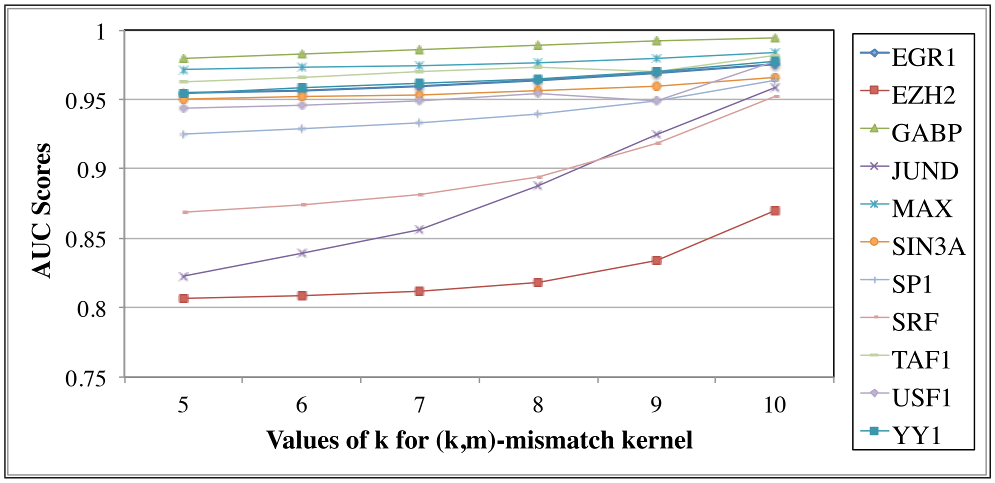


Figure 4.1: AUC Scores, for different Transcription Factors, when k is varied from 5 to 10 for the implemented (k, m)-mismatch string kernel (m=2). Prediction performance of SK increases with increase in k.

4.3 Effect of k parameter on the performance of SK

One of the most important hyperparameters to tune is k. Cross-validation based model selection is performed for choosing the hyperparameters, such as k. For selecting the k parameter, we decided to implement string kernel based model (SK). For example, Figure 4.1 shows the AUC scores with different values of k, where training was performed using data from GM12878 cell-type and test data was from K562 cells. This allowed us to understand what length of k-mers from the sequences, with mismatch $(m) = 2$, better captures the patterns among different sequences for our task. Figure 4.1 shows the prediction performance of using different k values for all the 11 selected TFs.

We observe an increase in the AUC scores (representing prediction performance) for all the TFs when k is increased from 5 to 10, while keeping m=2 constant in the (k, m)-mismatch kernel computation. Also, for TFs like EZH2, JUND and SRF, there is a steep increase in the performance with the increase of k. This indicates that their motifs are less consistent as compared to other TFs and need better techniques to capture motif patterns among their TFBS sequences. According to results in this case, one such strategy is to increase the value of k in the string kernel computation. Since overall $k = 10$ gives the best performance, rest of our analysis uses $(10, 2)$-mismatch kernel for kernel computation.

4.4 Performance Evaluation of SK versus Di-Mismatch Kernel

As mentioned earlier, we have used the “peak centric standard” for our evaluations. Nevertheless to compare our basic SK model to the di-mismatch kernel used in [1], we have relabeled our data according to the paper [1]. That is, our negative sequences were taken from the flanking regions 200 base-pairs away from the

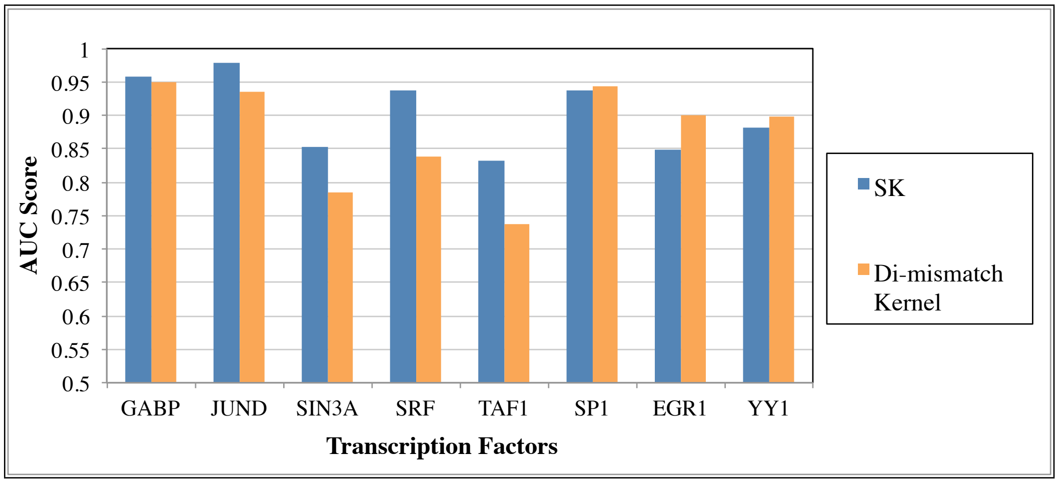


Figure 4.2: AUC Scores, for different Transcription Factors, generated by our SK model and that of di-mismatch kernel obtained from Table S3 in [1]. Our SK model performs better for most of the selected TFs.

center of the TF peak. Both models were trained on GM12878 datasets, while the predictions were made on TF data from K562 tissue. AUC scores for di-mismatch kernel of the common TFs were taken from the Supplementary section of [1]. The performance comparison of our baseline SK and di-Mismatch model is in Figure 4.2. Clearly, our basic SK model performs comparable or better than the di-mismatch kernel approach on the common TFs and using the labeling standard described in [1]. Having established that, we then we chose to perform further evaluations based on the "peak centric standard", where we treat random genomic regions without TFBS signal as negative sites. This is because we think that "peak-centric" standard better captures the real-world TFBS prediction scenario.

4.5 Performance Evaluation of SK versus TSK for Cross-Cell-type

Next, we implement the TSK approach where, after kernel computation of the training sequences, importance re-weighting is performed using weights calculated during Kernel Mean Matching step. In order to compare our TSK approach with the PWM-driven motif based approaches, we used the existing PWMs of TFs in JASPAR 2014 [7] database to score our test sequences for their TF binding potential. We used AMA tool [21] from the MEME suite [4] to calculate these scores. However, out of the 11 selected TFs, the motif based matrices for EZH2, SIN3A and TAF1 were not available in the database. Therefore, we present the prediction performance comparison between SK, TSK and PWM based models, for rest of the TFBS in Figure 4.3.

The AUC scores for TFs EZH2, SIN3A and TAF1, generated by SK and TSK are provided in Table 4.1. Figure 4.3 shows that TSK performs similar to or slightly better than SK for all the 11 TFBS. This indicates that there exists very little sample heterogeneity between TFBS sequences across these two cell

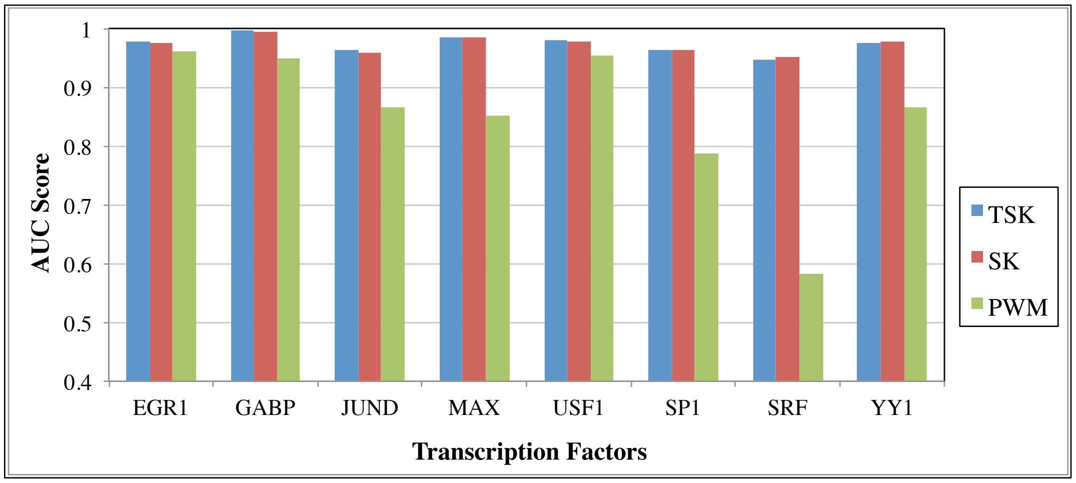


Figure 4.3: AUC Scores, for different Transcription Factors across different cell-types, generated by SK, TSK and PWM based approaches. TSK performs similar to or slightly better than SK for all TFs, while both SK and TSK outperform PWM based approach.

Table 4.1: AUC Scores for for TFs EZH2, SIN3A and TAF1, generated by SK and TSK. PWMs for these TFs were not available in the database. TSK performs similar to or slightly better than SK for selected TFs.

Transcription Factor	SK	TSK
EZH1	0.8694	0.8657
SIN3A	0.9655	0.9649
TAF1	0.9811	0.98392

types, with respect to the KMM re-weighting step. Another important observation, however, is that both the string kernel based methods (SK and TSK) perform better than traditional PWM-based approach. In cases like EZH2, SIN3A and TAF1, the availability of quality ChIP-seq data in at least one cell type will lead to successful TFBS predictions in any other cell type by using TSK. Therefore, TSK greatly reduces the needs to label context-specific TFBS through computational consideration of data heterogeneity and as well as data scarcity. As we have seen above, TSK based model is able to handle heterogeneity between TFBS datasets from different cell types. Next, we would like to check TSK's performance when facing the explicit data scarcity situation. Instead of training and testing the model on 1000 sequences, for GM12878 cell-type, we reduced the size of training datasets by half to simulate the scarcity case. Therefore the training data includes 500 positive and 500 negative sequences from GM12878 datasets while testing of the models was performed for 1000 TFBS positive and negative sequences of K562. We compare SK and TSK on this setting and report the results in Figure 4.4.

The aforementioned setting imitates the exact problem we aim to solve through transfer learning. That is, we are given samples aggregated from two different contexts, source and target, where labelled samples from source are sparse. Due to reduction of the training dataset by half, the prediction accuracy for some TFs

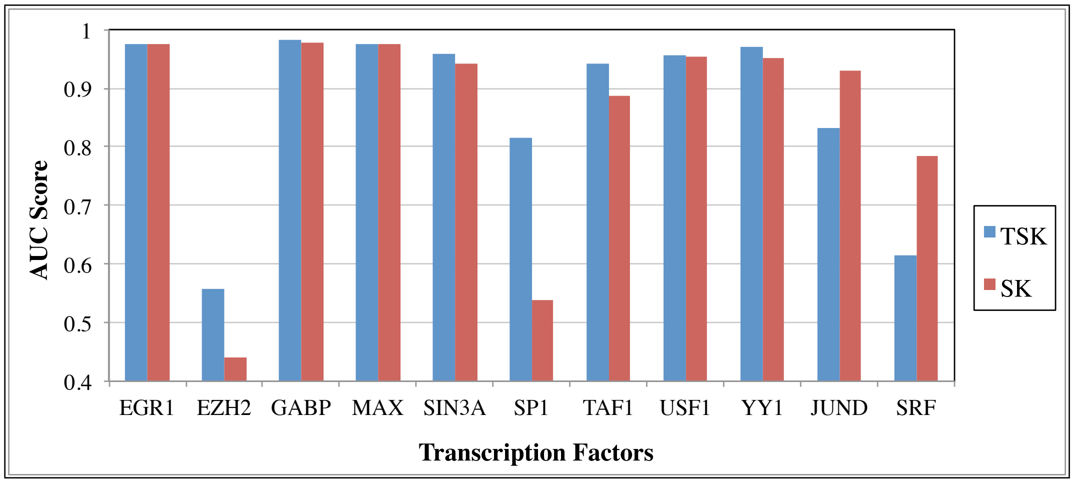


Figure 4.4: AUC Scores, for different Transcription Factors across different cell types with smaller training data, generated by SK and TSK based models. TSK performs better than the baseline string kernel (SK) based model for almost all the selected TFs, except JUND and SRF

reduces drastically, indicating that these TFs require more training examples to capture their motif patterns accurately. Even then, as expected, TSK performs better than the baseline string kernel (SK) based model for almost all the selected TFs . The only exception is being JUND and SRF, indicating that these TFs have the least heterogeneity among all the 11 TFs and thus, the sequence pattern can be accurately modeled by basic string kernel even if smaller training dataset is used.

4.6 Performance Evaluation of SK versus TSK for Cross-genome

We now demonstrate that our TSK approach can capture data heterogeneity and scarcity across other contexts, e.g. cross-genomes. We selected 10 TFs common between two related tissues (GM12878 and CH12) from human and mouse species.

The TSK approach has been implemented under 2 different variations: TSK and TSK'. The variation occurs during the importance re-weighting step before classification by SVM. "TSK" notation is used for the scenario, which consists of performing classification step on the kernel matrix, where features that are different and have been re-weighted with weights β are preserved, while rest of the features without weights are excluded. In the version with notation "TSK' ", features that are different and re-weighted like TSK' are kept, while at the same time, similar features are not discarded by assigning weights $\beta = 1$ to the rows of un-weighted features. This was done because it was observed that motifs of selected TFs were well-conserved among the cell-types of the two species, thus few features were preserved and weighted by TSK. Due to this, SVM did not have enough support vectors to perform accurate classification.

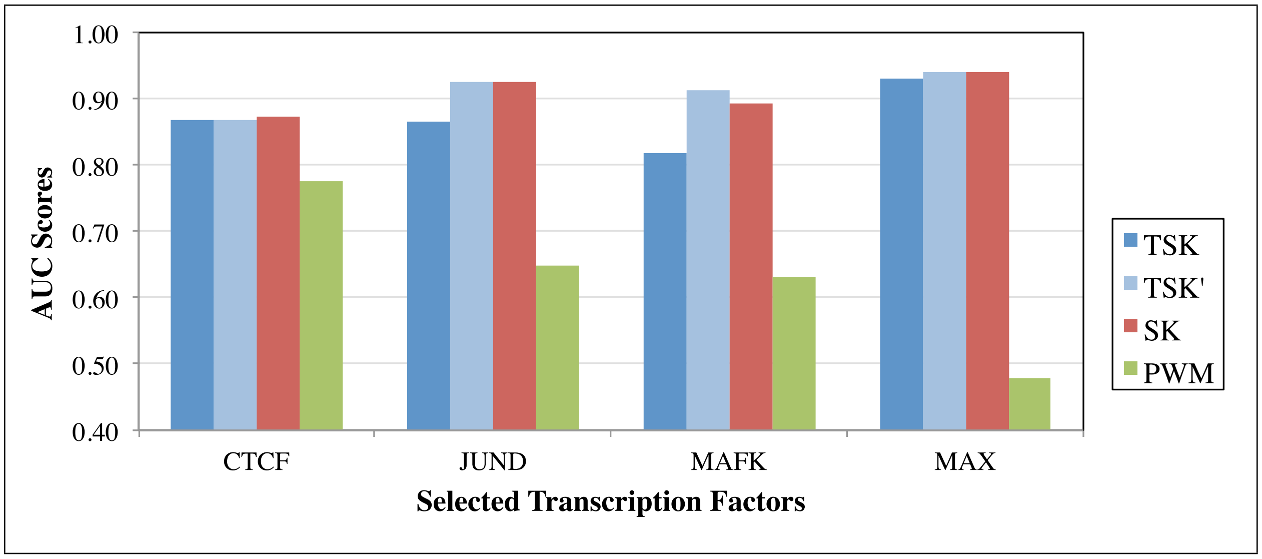


Figure 4.5: AUC values for SK, TSK based models (where former is baseline model and TSK and TSK' represent the two variations of importance re-weighting of the kernel matrix before classification) and PWM based methods. String kernel based approaches perform better than PWM based tools, while TSK' performs similar to SK for almost all the TFs except MAFK, where it outperforms SK.

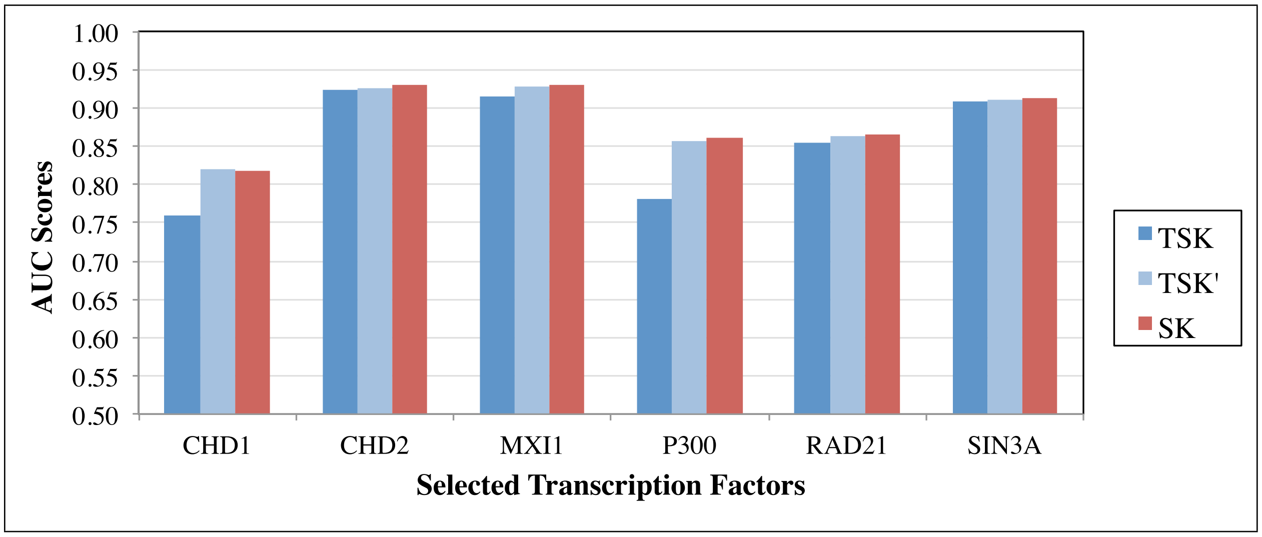


Figure 4.6: AUC values for SK versus TSK based models, where former is baseline model and TSK and TSK' represent the two variations of importance re-weighting of the kernel matrix before classification.TSK' performs similar to SK for almost all the TFs.

We once again, compared our string kernel based approach with PWM technique using the method described in previous section. This time, out of 10 selected TFs, only 4 TFs were available for PWM computation in JASPAR 2014 database [7]. The results for the 4 TFs are presented in Figure 4.5

Figure 4.6 shows comparison of TSK approaches, variations TSK and TSK', and SK for rest of the 6 TFs.

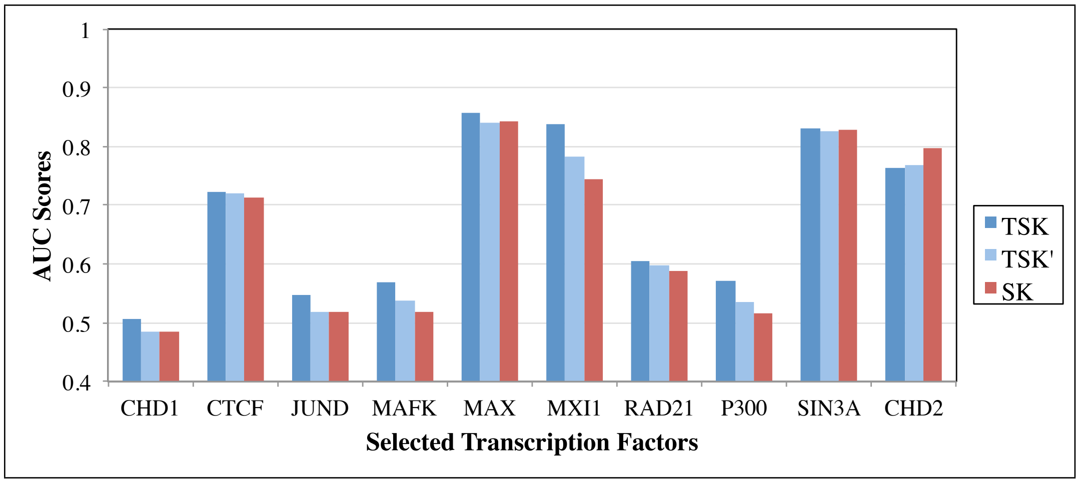


Figure 4.7: AUC Scores, for different Transcription Factors across cell types for different genomes with smaller training data, generated by SK and TSK based models. Both versions of TSK (TSK and TSK') perform better than the baseline string kernel (SK) based model for all the selected TFs, except CHD2.

It can be seen in Figure 4.5, both the string kernel based approaches outperform PWM based algorithm in scoring sequences for binding events. In Figures 4.5 and 4.6, we observe that TSK' performs similar to SK for almost all the TFs except MAFK, where it outperforms SK. This indicates that there is very little heterogeneity between other TFBS sequences across the cell types of the two species, for KMM re-weighting step to utilize. In case of MAFK, the motif might not be well preserved between human and mouse genomes and hence its prediction accuracy is helped by re-weighting the features based on differences in distribution densities. As discussed earlier, the lack of heterogeneity also leads to similar or lower AUC scores in case of simple TSK model as compared to SK and TSK' because of less number of weighted features for SVM to train on.

Using the exact same reduced training setup as the previous section, we applied both SK and TSK models for the training data comprising of 500 sequences from human TFBS data samples while testing the models on 1000 mouse TFBS sequences. Figure 4.7 presents the results for this comparison.

Similar as the cross-cell type results, both versions of TSK (TSK and TSK') perform better than the baseline string kernel (SK) based model for almost all the selected TFs . The only exception being CHD2, indicating that it has the least heterogeneity among all the 10 TFs and thus, the sequence motif similarity can be accurately modeled by basic string kernel even if smaller training dataset is used. In this case, because datasets are different in domains as well as sizes, most of the features are re-weighted with β values calculated during KMM re-weighting step. As a result, basic weighting scheme in TSK outperforms both SK and TSK' as SVM now has enough weighted features for calculating its support vectors.

The application of Transfer Learning enables us to transcend our training sets not only across different cell-types but also across different genomes. Our prediction model is not dependent on the availability of motif matrices or the availability of large amount of ChIP-seq data. Thus, using small number of available labelled ChIP-seq data in one context, confident predictions can be made about the TF binding event in a separate context. For a given set of regions, TSK can provide a ranked list based on probability of binding event at those locations.

Chapter 5

Related Work

Under a more general scenario, our framework is also related to the following relevant research topics.

5.1 Domain Adaptation

Domain Adaptation has been applied to a wide number of applications and has been proved to be an effective strategy. It utilizes the knowledge in a source domain to improve the learning tasks in a target domain, which has a different data distribution from the source. This clearly does not follow the underlying assumption of machine learning that normally assumes the training and testing data belong to similar feature space and are derived from same distribution. Such scenario could happen, for instance, the classification task may occur in one domain of interest, while there is lack of sufficient training data in another domain of interest. Furthermore, datasets from the two domains may belong to different feature space or follow different data distributions. For such scenarios, knowledge transfer, if done successfully, would greatly improve the performance of learning by avoiding the much expensive data-labeling efforts [15].

Consider a domain D consisting two components: a feature space χ and probability distribution $P(X)$, where $X = (x_1, \ldots, x_n) \in \chi$. Given a specific domain, $D = (\chi, P(X))$, a task consists of two components: a label space γ and an objective predictive function $f(.)$ (denoted by $T = (\gamma, f(.)))$, which is not observed but can be learned from the training data, which consist of pairs (x_i, y_i), where $x_i \in \chi$ and $y_i \in \gamma$. The function $f(.)$ can be used to predict the corresponding label, $f(x)$, of a new instance x. Source domain data can be denoted as $D_S = ((x_{S1}, y_{S1}), \ldots, (x_{Sn}, y_{Sn}))$, where $x_{Si} \in \chi_S$ is the data instance and $y_{Si} \in \gamma_S$ is the corresponding class label. Similarly, target-domain data can be denoted as $D_T = ((x_{T1}, y_{T1}), \ldots, (x_{Tn}, y_{Tn}))$, where the input $x_{Ti} \in \chi_T$ and $y_{Ti} \in \gamma_T$ is the corresponding output. In most cases, $0 \leq n_T \leq n_S$.

Thus, [15] formally defines Transfer Learning as:

Given a source domain D_S and learning task T_S, a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function $f(.)$ in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$.

5.2 Sequence Classification in Machine Learning

5.2.1 Generic Sequence Classification

A sequence is an ordered list of events. An event can be represented as a symbolic value, a numerical real value, a vector of real values or a complex data type. In [22], a *simple symbolic sequence* is defined as an ordered list of the symbols from the alphabet, given alphabet symbols $\{E_1, E_2, \ldots, E_n\}$.

For example, a DNA sequence is composed of 4 nucleotides A; C; G; T and a DNA segment, such as ATCGTAACGT, is a simple symbolic sequence.

A DNA sequence may belong to a TF binding site or non-binding site, thus the sequence carries a class label L. The task of a generic sequence classification is to learn a sequence classifier C, which is a function mapping a sequence s to a class label $l \in L$, written as, $C : s \rightarrow l; l \in L$. In conventional sequence classification, each sequence is associated with only one class label and the whole sequence is available to a classifier before the classification.

Generic sequence based classification methods can be divided into 3 categories [22] :

1. **Feature based classification**, which transforms a sequence into a feature vector and then applies conventional classification methods.

2. **Distance based classification**, where a distance function measures the similarity between sequences and determines the quality of the classification significantly.

3. **Model based classification**, such as using hidden markov model (HMM) and other statistical models to classify sequence.

5.2.2 Deep Learning based Sequence Classification

Recurrent Neural Networks (RNNs) have been used as a discriminative model where the output is a label sequence associated with the input data sequence [23]. Pre-segmented training data is usually required for training RNNs for discrimination. Post-processing is also needed to transform the generated outputs into label sequences. Work in [24] enables the RNNs themselves to perform sequence classification without the need to pre-segment training data or post-process output. The basic idea is to interpret RNN outputs as the conditional distributions over all possible label sequences given the input sequences. Next, a differentiable objective function can be derived to optimize these conditional distributions over the correct label sequences, such that no segmentation of data is required.

5.3 Weighted SVM

According to [25], the task of weighted support vector machine (WSVM) is to assign each data point a different weight according to its relative importance in the class such that different data point has different contribution to the learning of the decision surface. For a given weight, $0 < W_i < 1$, assigned to a data point x_i, the training datasets are:

$$\begin{aligned} \{(x_i, y_i, W_i)\}_{i=1}^{l}, \\ x_i \in R^N, \\ y_i \in \{-1, 1\}, \\ W_i \in R \end{aligned}$$

The WSVM tries to maximize the margin of separation and minimize the classification error such that a good generalization ability can be achieved. WSVM weighs the penalty term in order to reduce the affect of less important data points. This is different from SVM (discussed previously), where the value of C is fixed and all training data points are equally treated during the training process.

The constrained optimization problem can now be formulated as:

$$Minimize\phi(w) = \frac{1}{2}w^T w + C\sum_{i=1}^{l} W_i\xi \tag{5.1}$$

subject to

$$\begin{aligned} y_i(\langle w, \phi(x_i)\rangle + b) \geq 1 - \xi_i, i = 1, \ldots, l \\ \xi_i \geq 0, i = 1, \ldots, l \end{aligned}$$

As can be seen, weight W_i has been assigned to data point x_i in the above formulation. Now, the dual formulation becomes

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{5.2}$$

subject to

$$\sum_{i=1}^{l} y_i \alpha_i = 0,$$
$$0 \leq \alpha_i \leq CW_i, i = 1, \ldots, l$$

and the KKT conditions of WSVM become

$$\alpha_i [y_i(\langle w, \phi(x_i) \rangle + b) - 1 + \xi_i] = 0, i = 1, \ldots, l$$
$$(CW_i - \alpha_i)\xi_i = 0, i = 1, \ldots, l$$

Therefore, the sole difference between SVM and the WSVM is the upper bounds of Lagrange multipliers α_i in the dual problem. In SVM, the upper bounds of α_i are bounded by a constant C while they are bounded by dynamical boundaries that are weight values CW_i in WSVM.

Chapter 6

Conclusion

Determining how TF proteins interact with DNA to regulate gene expression is essential for fully understanding many biological processes and disease states. Most of the previous computational tools for predicting TF-DNA interactions assume similar distribution for target and source contexts. However, we have shown that it is possible to improve performance of the sequence based TFBS prediction models by considering the difference among underlying sample distributions and applying knowledge transfer. The insights obtained from applying TSK to cross-context TFBS applications could be summarized as follows:

- Both TSK and string kernel (SK) based models perform better than traditional PWM based approach.
- String kernel based approaches make it possible to be independent of availability of PWM databases.
- TSK performs similar to simple SK based approach when the data samples do not differ significantly from each other.
- With weights calculated by KMM, TSK is able to outperform SK by using samples from a different context.
- Using the resulting performance of TSK and SK, for TFBS predictions, we can categorize TFs into two groups, where one group has highly conserved binding sequence motifs while the other does not.

The results also establish that the value of k can affect the prediction accuracy of binding events of certain TFs drastically, thus the choice of parameters for the string kernel should be made carefully. From the discussion above, it is evident that Transfer String Kernel (TSK) is an efficient model. It can be applied to a wide variety of cross-context TFBS prediction tasks. It is able to handle both data distribution differences as well as data scarcity to provide an accurate model, which is not biased towards any underlying assumptions of similarity between datasets. TSK not only performs better than existing tools but is also independent of the source of TFBS sequence data or the size of labelled training samples.

6.1 Future Work

For future work, we plan to convert our TSK model into a tool, so that Biologists are able to make predictions of a TF binding event on regions of interest for any cell type of any genome. We also plan to incorporate other types of available features related to these TFBS, like histone modifications, DNAse-seq data etc., through multiple-kernel learning, thus providing a comprehensive understanding of the TF binding event and its relationship with other simultaneously occurring changes in the genomic environment. The proposed extensions to the existing model are possible due to its flexibility. Next step also includes application of deep-learning model, to better handle the different types of input features and enable multi-class classification, where we not only separate sequences into binding or non-binding regions but also flanking regions. Thus, TSK provides a framework to further improve the performances of state-of-the-art models designed for the task of TFBS predictions across cell-types, cell-environments as well as genomes.

Bibliography

[1] Aaron Arvey, Phaedra Agius, William Stafford Noble, and Christina Leslie. Sequence and chromatin determinants of cell-type–specific transcription factor binding. *Genome research*, 22(9):1723–1734, 2012.

[2] Peter J Park. Chip–seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics*, 10(10):669–680, 2009.

[3] Ivan V Kulakovskiy, VA Boeva, Alexander V Favorov, and VJ Makeev. Deep and wide digging for binding motifs in chip-seq data. *Bioinformatics*, 26(20):2622–2623, 2010.

[4] Timothy L Bailey, Mikael Boden, Fabian A Buske, Martin Frith, Charles E Grant, Luca Clementi, Jingyuan Ren, Wilfred W Li, and William S Noble. Meme suite: tools for motif discovery and searching. *Nucleic acids research*, 37(suppl 2):W202–W208, 2009.

[5] Hongzhu Qu and Xiangdong Fang. A brief review on the human encyclopedia of dna elements (encode) project. *Genomics, proteomics & bioinformatics*, 11(3):135–141, 2013.

[6] ENCODE Project Consortium et al. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57–74, 2012.

[7] Anthony Mathelier, Xiaobei Zhao, Allen W Zhang, François Parcy, Rebecca Worsley-Hunt, David J Arenillas, Sorana Buchman, Chih-yu Chen, Alice Chou, Hans Ienasescu, et al. Jaspar 2014: an extensively expanded and updated open-access database of transcription factor binding profiles. *Nucleic acids research*, page gkt997, 2013.

[8] Matthew T Weirauch, Atina Cote, Raquel Norel, Matti Annala, Yue Zhao, Todd R Riley, Julio Saez-Rodriguez, Thomas Cokelaer, Anastasia Vedenko, Shaheynoor Talukder, et al. Evaluation of methods for modeling transcription factor sequence specificity. *Nature biotechnology*, 31(2):126–134, 2013.

[9] Christina S. Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for svm protein classification. In *Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[10] Christina S. Leslie, Eleazar Eskin, Jason Weston, and William Stafford Noble. Mismatch string kernels for svm protein classification. In *NIPS*, pages 1417–1424, 2002.

[11] Pavel P Kuksa, Pai-Hsi Huang, and Vladimir Pavlovic. Scalable algorithms for string kernels with inexact matching. In *NIPS*, pages 881–888, 2008.

[12] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[13] Christina Leslie and Rui Kuang. Fast string kernels using inexact matching for protein sequences. *The Journal of Machine Learning Research*, 5:1435–1455, 2004.

[14] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998.

[15] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[16] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

[17] Yao-liang Yu and Csaba Szepesvári. Analysis of kernel mean matching under covariate shift. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 607–614, 2012.

[18] Yong Lu, Peter Huggins, and Ziv Bar-Joseph. Cross species analysis of microarray expression data. *Bioinformatics*, 25(12):1476–1483, 2009.

[19] Gabriel Cuellar-Partida, Fabian A Buske, Robert C McLeay, Tom Whitington, William Stafford Noble, and Timothy L Bailey. Epigenetic priors for identifying active transcription factor binding sites. *Bioinformatics*, 28(1):56–62, 2012.

[20] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel methods in computational biology*. MIT press, 2004.

[21] Fabian A Buske, Mikael Bodén, Denis C Bauer, and Timothy L Bailey. Assigning roles to dna regulatory motifs using comparative genomics. *Bioinformatics*, 26(7):860–866, 2010.

[22] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1):40–48, 2010.

[23] Li Deng. Three classes of deep learning architectures and their applications: A tutorial survey.

[24] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[25] Xulei Yang, Qing Song, and Yue Wang. A weighted support vector machine for data classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(05):961–976, 2007.