Journal of Machine Learning Research 10 (2009) 1913-1936 Submitted 10/08; Revised 3/09; Published 8/09

Learning Approximate Sequential Patterns for Classification

Zeeshan Syed ZHS@EECS.UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2121, USA

Piotr Indyk INDYK@CSAIL.MIT.EDU
John Guttag GUTTAG@CSAIL.MIT.EDU
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139-4307, USA

Editor: Charles Elkan

Abstract

In this paper, we present an automated approach to discover patterns that can distinguish between sequences belonging to different labeled groups. Our method searches for approximately conserved motifs that occur with varying statistical properties in positive and negative training examples. We propose a two-step process to discover such patterns. Using locality sensitive hashing (LSH), we first estimate the frequency of all subsequences and their approximate matches within a given Hamming radius in labeled examples. The discriminative ability of each pattern is then assessed from the estimated frequencies by concordance and rank sum testing. The use of LSH to identify approximate matches for each candidate pattern helps reduce the runtime of our method. Space requirements are reduced by decomposing the search problem into an iterative method that uses a single LSH table in memory. We propose two further optimizations to the search for discriminative patterns. Clustering with redundancy based on a 2-approximate solution of the *k*-center problem decreases the number of overlapping approximate groups while providing exhaustive coverage of the search space. Sequential statistical methods allow the search process to use data from only as many training examples as are needed to assess significance. We evaluated our algorithm on data sets from different applications to discover sequential patterns for classification. On nucleotide sequences from the Drosophila genome compared with random background sequences, our method was able to discover approximate binding sites that were preserved upstream of genes. We observed a similar result in experiments on ChIP-on-chip data. For cardiovascular data from patients admitted with acute coronary syndromes, our pattern discovery approach identified approximately conserved sequences of morphology variations that were predictive of future death in a test population. Our data showed that the use of LSH, clustering, and sequential statistics improved the running time of the search algorithm by an order of magnitude without any noticeable effect on accuracy. These results suggest that our methods may allow for an unsupervised approach to efficiently learn interesting dissimilarities between positive and negative examples that may have a functional role.

Keywords: pattern discovery, motif discovery, locality sensitive hashing, classification

1. Introduction

Pattern discovery has been studied extensively in the context of data mining and knowledge discovery (Han and Kamber, 2005) and causal inference in statistics (Pearl, 2000). The search for patterns is typically guided by classification. The focus is on discovering activity that can distinguish members of a family from non-members (Duda et al., 2000) by identifying activity that is unlikely to occur purely by chance and may have a functional role (Syed et al., 2007).

Pattern discovery has been applied to data from a variety of applications, for example, world wide web transactions (Mobasher et al., 1996), marketing information (Shaw et al., 2001), and medical signals (Li et al., 2005). More recently, there has been an increased interest in applying techniques for discovering patterns to sequences corresponding to genomic data (Wang et al., 1999). Of particular importance in computational biology is the problem of discovering subsequences, that is, motifs, which regulate important biological processes (Kellis et al., 2004). Pattern discovery has been proposed in this context as a machine learning problem (Brazma et al., 1998):

> Given two sets of sequences S^+ and S^- drawn randomly from families F^+ and F^- respectively such that $F^+ \cap F^- = \oslash$, find the pattern W of length L that has high likelihood in F^+ but not in F^-.

This formulation is sufficiently general to apply to a wide variety of applications where sequential data exists. Furthermore, an extensive literature on symbolization (Daw et al., 2003) allows for a large set of time-series signals to be abstracted into sequential data for analysis. We make the notion of a pattern more explicit by refining the goal of pattern discovery described above as follows:

> Given two sets of sequences S^+ and S^- drawn randomly from families F^+ and F^- respectively such that $F^+ \cap F^- = \oslash$, find the subsequence W of length L that occurs with a Hamming distance of at most d with high likelihood in F^+ but not in F^-.

In this paper, we propose a method to efficiently carry out the search for such approximate patterns. A variety of techniques have been proposed to address this problem statement (Lawrence et al., 1993; Bailey and Elkan, 1994; Grundy et al., 1997; Tavazoie et al., 1999; Liu et al., 2001; Pavesi et al., 2001; Sinha and Tompa, 2003). The common strategy adopted by these methods is to approach the problem of pattern discovery by finding activity that is statistically unlikely but occurs consistently in positive examples. Negative examples are primarily used for evaluation. This process means that discriminative patterns in negatively labeled sequences are not explored for classification. Other algorithms for pattern discovery (Delcher et al., 1999; Batzoglou et al., 2000) enumerate all exact patterns across both positive and negative examples to identify sequences that can discriminate between these two cases, but become computationally intractable when allowing subsequences to have an approximate form.

We describe a locality sensitive hashing (LSH) based algorithm to efficiently estimate the frequencies of all approximately conserved subsequences with a certain Hamming radius in both positive and negative examples. The search process attempts to identify patterns that allow maximum discrimination between the two groups. In this way, our method unifies the broad areas of existing work in sequential pattern detection for classification by proposing a way to discover patterns that are both approximate and derived using the additional information available in negative instances.

LSH forms a key component of our method. The use of LSH has been proposed earlier in the context of pattern discovery to identify interesting activity in positive examples (Buhler, 2001; Buhler and Tompa, 2002). We supplement this work by allowing for information from negative examples to be factored into the search and by proposing different optimizations to the search process. In particular, we expand the use of LSH in pattern discovery from indexing to fast counting and approximate clustering. While LSH provides runtime efficiency to the search process, it imposes significant space requirements, and we describe an iterative method that uses a single LSH table in memory to address this issue. We also explore the idea of using clustering as part of pattern discovery to reduce approximate subsequences with significantly overlapping Hamming radii to a small number. This aspect of our work resembles efforts for web clustering (Haveliwala et al., 2000). We explore similar ideas within the context of approximate pattern discovery. This decreases the number of motifs to be evaluated while still providing a fairly exhaustive coverage of the search space. We describe a clustering method based on a 2-approximate solution of the k-center problem to achieve this goal.

In addition to LSH and clustering, we also draw upon sequential statistical methods to make the search for interesting patterns more efficient. The process of identifying patterns with discriminative ability makes use of concordance and rank sum testing. In many cases, the goodness of approximate patterns can be assessed without using data from all training sequences. We propose a further optimization to address these cases. The runtime and space requirements of the pattern discovery process can be reduced by using sequential statistical methods that allow the search process for patterns to terminate after using data from only as many training examples as are needed to assess significance.

We address a similar goal to earlier work on using hypergeometric significance testing to discover patterns that are enriched in a positive set relative to a negative set (Barash et al., 2001). The focus of this work is to generate seeds of short lengths that can be expanded using an expectation-maximization (EM)-like process to produce a position specific scoring matrix of the desired length. However, in contrast to our work, this method is based on the assumption that a pattern occurs at most once in each sequence. This leads it to disregard multiple copies of a match within the same sequence. Moreover, the use of a testing function based on hypergeometric analysis may affect the accuracy of this method (Leung and Chin, 2006).

Our algorithm to find approximate discriminative patterns is also related to previous work on the use of profile hidden Markov models (Krogh, 1994; Jaakkola et al., 1999) to optimize recognition of positively and negatively labeled sequences. This work focuses on learning the parameters of a hidden Markov model that can represent approximations of subsequences. Generally, this approach requires large amounts of data or sophisticated priors to train the hidden Markov model. Computing forward and backward probabilities from the Baum-Welch algorithm is also very computationally intensive. Subsequent work in this area focuses on mismatch tree-based kernels (Leslie et al., 2003) for use in a support vector machine (SVM) classifier. This work focuses on efficiently calculating a kernel based on the mismatch tree data structure (Eskin and Pevzner, 2002), which quantifies how different two sequences are based on the approximate occurrence of the fixed L-length subsequences within them. The mismatch kernel is used to train an SVM and assign labels to unknown query sequences.

Our algorithm supplements this work by measuring how frequently each subsequence occurs in an approximate form in the data. In contrast to the mismatch kernel, which focuses on quantifying the difference between two sequences and does not report the frequency of individual approximate

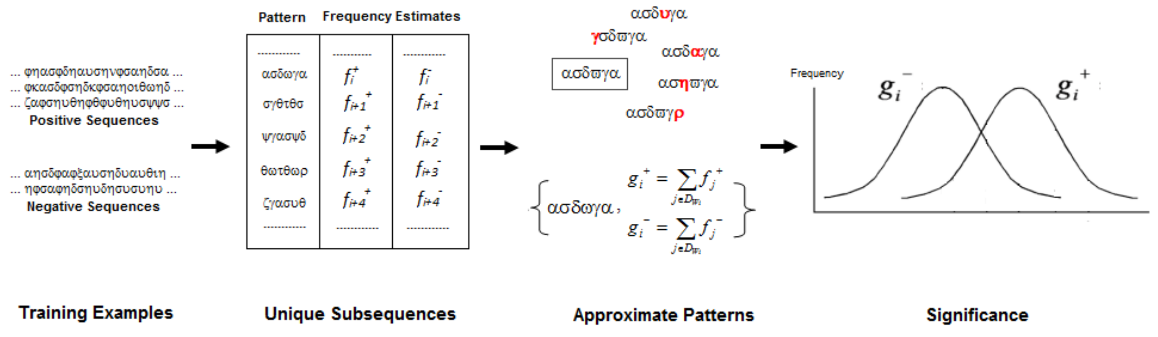

Figure 1: Overview of the pattern discovery process.

subsequences in the data, our algorithm focuses on identifying the specific approximate patterns with discriminative value. This approach can be integrated more easily with the use of sequential statistics, that is, since the frequencies of each approximate pattern are retained during analysis, this information can be used to determine if patterns are good or bad discriminators without analyzing all the available data.

We evaluated our algorithm on data sets from different applications to discover sequential patterns for classification. On nucleotide sequences from the Drosophila genome, our method was able to discover binding sites for genes that are preserved across the genome and do not occur in random background sequences. On symbolized electrocardiographic time-series from patients with acute coronary syndromes, our pattern discovery approach identified approximately conserved sequences of morphology variations that were predictive of future death in a test population. These results suggest that our methods may allow for an unsupervised approach to learn interesting dissimilarities between positive and negative examples that may have a functional role.

The rest of this paper is organized as follows: Section 2 gives an overview of our algorithm. Section 3 describes a locality sensitive hashing scheme to find approximate matches to all subsequences in the data set. Section 4 proposes the use of clustering to reduce the number of approximate patterns analyzed during pattern discovery. Section 5 discusses the statistical approaches used in assessing the goodness of patterns. Section 6 details the evaluation methodology of our pattern discovery algorithm on data from different real-world applications. Section 7 reports the results of this study. Section 8 concludes with a discussion.

2. Overview

The process of discovering discriminative patterns of a specified length L from positive and negative sequences is carried out in two stages: frequency estimation and pattern ranking. Figure 1 presents an overview of this approach.

2.1 Frequency Estimation

Given a set of positive examples $S^+ = \{S_x^+ | x = 1,\ldots,N^+\}$ and a set of negative examples $S^- = \{S_y^- | y = 1,\ldots,N^-\}$ the frequency estimation step measures the frequency of every unique subsequence W_i for $i = 1, \ldots, M$ in sequences belonging to S^+ and S^-. The resulting frequency for W_i in positive and negative examples is denoted as:

$$f_i^+ = \{f_{i,z}^+ | z \in S^+\},$$
$$f_i^- = \{f_{i,z}^- | z \in S^-\}$$

where $f_{i,z}^+$ and $f_{i,z}^-$ are the frequencies with which W_i appears in sequences z drawn from S^+ and S^-, and f_i^+ and f_i^- are vectors measuring the frequency of W_i in all positive and negative sequences.

To allow for approximate patterns, unique subsequences are then matched to all other subsequences at a Hamming distance of at most d from W_i. Denoting this group of subsequences as D_{W_i}, the resulting frequency for the subsequence W_i and its approximate matches is defined as:

$$g_i^+ = \sum_{j \in D_{W_i}} f_j^+,$$
$$g_i^- = \sum_{j \in D_{W_i}} f_j^-$$

where g_i^+ and g_i^- are vectors obtained by summing up the vectors f_i^+ and f_i^- for all subsequences within a given Hamming radius d of W_i.

In Section 3, we describe an LSH-based solution that allows for efficient discovery of the subsequences D_{W_i} matching W_i. We also present a clustering approach in Section 4 to reduce overlapping approximate patterns for which frequencies are estimated to a smaller number with less redundancy for subsequent analysis.

2.2 Pattern Ranking

The goal of the search process is to identify approximately matching subsequences that can discriminate between positive and negative training examples. The pattern ranking stage therefore scores each candidate approximate pattern according to its discriminative ability. We use two measures to assess the goodness of patterns.

The first approach to score patterns is to use rank sum testing. This technique is a non-parametric approach for assessing whether two samples of observations come from the same distribution. Patterns are ordered based on the significance of separation (as measured by the p-value) obtained by rank sum testing. A second scoring criterion used by our work is the C-statistic, which corresponds to the area under the receiver operating characteristic (ROC) curve. Details of these techniques are provided in Section 5. We further describe how sequential methods can be used to reduce the search process to only process as many training examples as are needed to determine if a candidate pattern has high or low discriminative ability.

3. Locality Sensitive Hashing

In this section, we describe the use of locality sensitive hashing in our algorithm.

3.1 Finding Approximate Matches for a Subsequence

Locality sensitive hashing (Indyk and Motwani, 1998) has been proposed as a randomized approximation algorithm to solve the nearest neighbor problem. Given a set of subsequences, the goal of

LSH is to pre-process the data so that future queries searching for closest points under some l_p norm can be answered efficiently. A brief review of LSH is presented here.

Given two subsequences S_x and S_y of length L, we describe them as being similar if they have a Hamming distance of at most d. To detect similarity, we choose K indices $i_1, \ldots, i_K$ at random with replacement from the set $\{1, \ldots, L\}$. The locality sensitive hash function $LSH(S)$ is then defined as:

$$LSH(S) = < S[i_1], \ldots, S[i_k] >$$

where $< \ldots >$ corresponds to the concatenation operator. Under this scheme, S_x and S_y are declared to be similar if:

$$LSH(S_x) = LSH(S_y). \tag{1}$$

The equality in Equation 1 corresponds to an exact match. However, since the indices used by the locality sensitive hash function $LSH(S)$ may not span the entire subsequences S_x and S_y, an exact match in Equation 1 may be obtained if S_x and S_y match approximately.

Practically, LSH is implemented by creating a hash table using the $LSH(S)$ values for all subsequences as the keys. Searching for the approximate neighbors of a query subsequence corresponds to a two-step process. The locality sensitive hash function is first applied to the query. Following this, the bucket to which the query is mapped is searched for all original subsequences with a Hamming distance of at most d.

Two subsequences with a Hamming distance of d or less may not match for a random choice of K indices if one of the K indices chosen corresponds to a position in which S_x and S_y differ. The probability of such a miss is bounded by (Indyk and Motwani, 1998):

$$Pr[LSH(S_x) \neq LSH(S_y)] \leq [1 - (1 - \frac{d}{L})^K].$$

By repeating the process of choosing K indices T times this probability can be reduced further to:

$$Pr[LSH(S_x) \neq LSH(S_y)] \leq [1 - (1 - \frac{d}{L})^K]^T. \tag{2}$$

Effectively, Equation 2 corresponds to constructing a data structure comprising T hash tables using different locality sensitive hash functions $LSH_1(S), \ldots, LSH_T(S)$. Approximate neighbors for a query are detected by searching for matches in each of these hash tables as described earlier.

The intuition underlying LSH is that the problem of searching through all possible subsequences in the data set for a match can be reduced to the more feasible problem of first rapidly identifying a small set of potential matches with a bounded error, and then searching through this smaller set to remove false positives. The lower the desired error bound for false negatives affecting correctness (i.e., by choosing K and T), the higher the corresponding false positive rate affecting the runtime of the algorithm. The choice between these two parameters depends on the application and the underlying data set.

3.2 Finding Approximate Matches Between All Subsequences

LSH provides an efficient mechanism to find the nearest neighbors of a given subsequence. To find the nearest neighbors for all M subsequences in the data set, each member of the set can be passed through the entire LSH data structure comprising T hash tables for matches. Unfortunately, this process is both computationally and memory intensive. In what follows, we describe a strategy to reduce the space requirements of LSH-based search for all approximate matches between subsequences. Section 4 further addresses runtime issues by proposing a clustering amendment to the search process.

Different approaches have been proposed recently to reduce the space requirements of LSH. In particular, the use of multi-probe LSH (Lv et al., 2007) has been shown to substantially reduce the memory requirements for traditional LSH by searching each LSH hash table (i.e., corresponding to a random selection of K indices) more thoroughly for misses. This additional work translates into fewer LSH hash tables being needed to bound the given error rate. As a result, the space of the LSH data structure decreases.

In our work, the memory requirements of LSH are reduced by organizing the approximate matching process as T iterations. Each iteration makes use of a single locality sensitive hash function and maintains only a single hash table in memory at any time. To preserve state across iterations, the search process maintains a list of matching pairs found during each loop after removing false positives. The subsequences D_{W_i} matching W_i are found as:

$$D_{W_i} = \bigcup_{t=1}^{T} \{W_j | LSH_t(W_j) = LSH_t(W_i)\}.$$

4. Clustering Subsequences

The runtime of the pattern discovery process as described so far is dominated by the approximate matching of all subsequences. Every subsequence is first used to create the LSH data structure, and then passed through the LSH data structure to find matches with a Hamming distance of at most d. This process is associated with considerable redundancy, as matches are sought individually for subsequences that are similar to each other. The overlap between approximate patterns increases the computational needs of the pattern discovery process and also makes it more challenging to interpret the results as good patterns may appear many times in the output.

To address this issue, we reduce patterns to a much smaller group that still collectively spans the search space. This is done by making use of a clustering method based on a 2-approximate solution to the k-center problem. The focus of this clustering is to group together the original subsequences falling into the same hash bucket during the first LSH iteration. Each of the clusters obtained at the end of this process corresponds to an approximate pattern that is retained. During subsequent iterations, while all subsequences are still used to construct the LSH tables, only the cluster centroids are passed through the LSH data structure. This reduces the runtime of the search by reducing the number of times subsequences have to be passed through the LSH tables to find true and false positives. It also reduces the memory requirements of the search by reducing the number of subsequences for which we need to maintain state about approximate matches.

The traditional k-center problem can be formally posed as follows. Given a complete graph $G = (V, E)$ with edge weights $\omega_e \geq 0$, $e \in E$ and $\omega(v, v) = 0$, $v \in V$, the k-center problem is to find a subset $Z \in V$ of size at most k such that the following quantity is minimized:

$$W(Z) = \max_{i \in V} \min_{j \in Z} \omega_{(i,j)}.. \tag{3}$$

The k-center problem is NP-hard, but a 2-approximate solution has been proposed (Hochbaum and Shmoys, 1985) for the case where the triangular inequality holds:

$$\omega_{(i,j)} + \omega_{(j,k)} \geq \omega_{(i,k)}.$$

The Hamming distance metric obeys the triangular inequality. Under this condition, the process of clustering can be decomposed into two stages. During the first LSH iteration, we identify subsequences that serve as cluster seeds using the 2-approximate solution to the k-center problem. Subsequent LSH iterations are used to grow the clusters till the probability that any subsequence within a Hamming distance at most d of the cluster centroid is missed becomes small. This approach can be considered as being identical to choosing a set of subsequences during the first LSH iteration, and finding their approximate matches by multiple LSH iterations.

More formally, during the first LSH iteration, for each bucket b_i in the hash table for $i = 1, \ldots, B$, we solve the k-center problem using the 2-approximate method (Hochbaum and Shmoys, 1985) with a Hamming distance metric. The number of subsequences forming centers k_i for the i-th hash table bucket is determined alongside the specific centroid subsequences from:

$$k_i = \min\{k | W(z_i(k)) \leq d\} \tag{4}$$

where $W(Z)$ is defined as in Equation 3 and $z_{i(k)}$ denotes the subsequence centers chosen for a particular choice of k in Equation 4, that is:

$$k_i = \min\{k | \max_{j \in b_i} \min_{z_i(k)} \omega_{(j, z_i(k))} \leq d\}.$$

The final set of subsequences chosen as centroids at the end of the first LSH iteration then corresponds to:

$$\Phi = \bigcup_{i=1}^{B} z_i(k_i).$$

The LSH iterations that follow find approximate matches to the subsequences in Φ. It is important to note that while clustering reduces a large number of overlapping approximate patterns to a much smaller group, the clusters formed during this process may still overlap. This overlap corresponds to missed approximate matches that do not hash to a single bucket during the first LSH iteration. Techniques to merge clusters can be used at the end of the first LSH iteration to reduce overlap. In our work, we tolerate small amounts of overlap between clusters analogous to the use of sliding windows to more thoroughly span the search space. Figure 2 illustrates the clustering process.

5. Pattern Ranking

Given the frequencies g_i^+ and g_i^- of an approximate pattern corresponding to all subsequences within a Hamming distance d of the subsequence W_i, a score can be assigned to the pattern by using concordance statistics and rank sum testing.

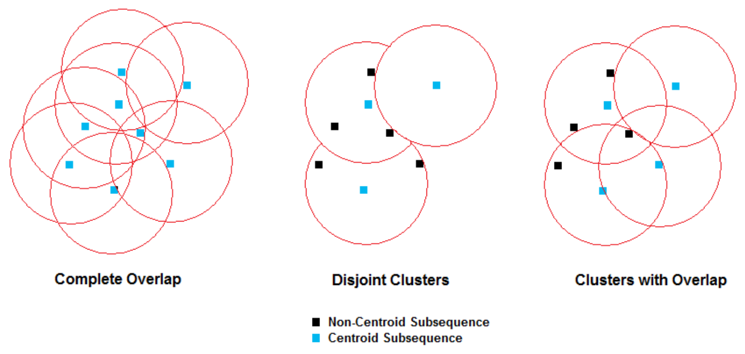

Figure 2: In the absence of clustering there is significant redundancy between the Hamming radii of approximate patterns. Partitioning the data into disjoint clusters can help address this issue. In our work, we reduce the original approximate patterns into a small group with some overlap to span the search space.

5.1 Concordance Statistic

The concordance statistic (C-statistic) (Hanley and McNeil, 1982) measures the discriminative ability of a feature to classify binary endpoints. The C-statistic corresponds to the area under the receiver operating characteristic (ROC) curve, which describes the inherent trade-off between sensitivity and specificity. As opposed to measuring the performance of a particular classifier, the C-statistic directly measures the goodness of a feature (in this case the frequency with which an approximate pattern occurs) by evaluating its average sensitivity over all possible specificities.

The C-statistic ranges from 0-1. A pattern that is randomly associated with the labels would have a C-statistic of 0.5. Conversely, good discriminators would correspond to either low or high C-statistic values.

5.2 Rank Sum Testing

An alternate approach to assess the goodness of patterns is to make use of rank sum testing (Wilcoxon, 1945; Lehmann, 1975). This corresponds to a non-parametric method to test whether a pattern occurs with statistically different frequencies in both positive and negative examples.

Given the frequencies g_i^+ and g_i^- of an approximate pattern in both positive and negative examples, the null and alternate hypotheses correspond to:

$$H_0 : \mu_{g_i^+} = \mu_{g_i^-},$$
$$H_1 : \mu_{g_i^+} \neq \mu_{g_i^-}.$$

Rank sum testing calculates the statistic U whose distribution under H_0 is known. This is done by arranging the g_i^+ and g_i^- into a single ranked series. The ranks for the observations from the g_i^+ series are added up. Denoting this value as R^+ and the number of positive examples by N^+, the statistic U is given by:

$$U = R^+ - \frac{N^+(N^+ + 1)}{2}.$$

The obtained value of U is compared to the known distribution under H_0 and a probability for this observation corresponding to the null hypothesis is obtained (i.e., the p-value). For large samples, U is approximately normally distributed and its standardized value can be checked in tables of the normal distribution (Gibbons, 1985; Hollander and Wolfe, 1999). The lower the p-value obtained through rank sum testing, the more probable it is that g_i^+ and g_i^- have distributions with different means, that is, that the approximate pattern is distributed differently in positive and negative examples.

5.3 Sequential Statistical Tests

The runtime and space requirements of the pattern discovery process can be reduced by analyzing only a subset of the training data. For example, it may be possible to recognize patterns with high discriminative ability without the need to analyze all positive and negative examples.

Our pattern discovery algorithm starts out by using a small subset of the initial training data for batch analysis. This helps identify candidate approximate patterns that occur in the data set and collectively span the search space. The remaining training examples are used for scoring purposes only. These examples are added in an online manner and during each iteration, the occurrence of outstanding candidate patterns in positive and negative examples is updated. Patterns may then be marked as being good or bad (and consequently removed from further analysis) or studied further to resolve uncertainty. The number of candidate patterns is therefore monotonically non-increasing with iteration number. As a result, the analysis of training examples becomes faster as more data is added, since fewer patterns need to be scored.

A sequential formulation for rank sum testing has been proposed (Phatarfod and Sudbury, 1988) that adds positive and negative examples in pairs. The frequencies of an approximate pattern in positive and negative examples at the end of iteration n can be denoted as $g_i^+(j)$ and $g_i^-(j)$ where $j = 1, \ldots, n$. The corresponding statistic for rank sum testing is:

$$U_n = \sum_{x=1}^{n} \sum_{y=1}^{n} I(g_i^+(x) > g_i^-(y)). \tag{5}$$

The operator $I(.)$ is equal to one when the inequality holds and zero otherwise. Using this statistic, the decision at the end of the n-th iteration corresponds to accepting H_0 if:

$$\frac{U_n}{n} < \frac{n}{2} - \lambda \log(\frac{1-\beta}{\alpha}) \tag{6}$$

while H_1 is accepted if:

$$\frac{U_n}{n} > \frac{n}{2} - \lambda \log(\frac{\beta}{1-\alpha})$$

where λ is defined as Phatarfod and Sudbury (1988):

$$\lambda = \frac{1 - \frac{\delta^2}{2} - \frac{\delta^3}{3\sqrt{3}} - \frac{\delta^4}{48}}{2\sqrt{3}\delta - \frac{\delta^2}{2}}. \tag{7}$$

In Equations 6 to 7, α and β correspond to desired false positive and false negative rates for the sequential rank sum testing process, while δ is a user-specified parameter that reflects the preferences of the user regarding how fine or coarse a difference the distributions are allowed to have under the alternate hypothesis. If both inequalities are not met, the process of adding data continues until there are no more training examples to add. In this case, all outstanding candidate patterns are rejected.

This formulation of sequential rank sum testing adds data in pairs of positive and negative examples. In cases where there is a skew in the training examples (without loss of generalization we assume a much larger number of negative examples than positive ones), we use a different formulation of sequential testing (Reynolds, 1975). Denoting the mean frequency in positive training samples as:

$$\mu_i^+ = \sum_{x=1}^{N^+} g_i^+(x).. \tag{8}$$

The alternate hypothesis can be redefined as the case where $h_i = g_i^- - \mu_i^+$ is asymmetrically distributed about the origin. This can be identified using the statistic:

$$U_n = \sum_{x=1}^{n} \frac{1}{x+1} \operatorname{sgn}(h_i(x)) R_{xx} \tag{9}$$

where R_{xy} is defined as the rank of $|h_i(x)|$ in the set $\{|h_i(1)|, \ldots, |h_i(y)|\}$ with $x \leq y$ and $\operatorname{sgn}(|h_i(x)|)$ is 1 if $h_i(x) \geq 0$ and -1 otherwise. The test procedure using the statistic continues taking observations as long as $U_n \in (-\delta, \delta)$ where δ is a user-specified parameter.

Traditional formulations of sequential significance testing remove good patterns from analysis when the test statistic is first found to lie above or below a given threshold. Patterns with potentially low discriminative ability are retained for further analysis and are discarded if they do not meet any of the admission criteria during any iteration of the search process. Since there may be a large number of such patterns with low discriminative value, we make use of a modified approach to reject poor hypotheses while retaining patterns that may have value in classification. This strategy improves efficiency, while also ensuring that good patterns are ranked using the available training data. Given the typical goal of pattern discovery to return the best patterns found during the search process, this technique naturally addresses the problem statement.

Given the test statistics in Equations 5 and 9, we remove all patterns that have a test statistic:

$$U_n < \lambda U_{\max}$$

where $U_{\max}$ is the maximum test statistic for any pattern and λ is a user-specific fraction (e.g., 0.2).

5.4 Multiple Hypothesis Correction

While assessing a large number of approximate patterns, M, the statistical significance required for goodness must be adjusted for Type I (i.e., false positive) errors. If we declare a pattern to be significant for some probability of the null hypothesis less than θ, then the overall false positive rate for the experiment assuming independence of patterns is given by:

$$FP = 1 - (1 - \theta)^M.$$

If we do not assume that the patterns are independent, the false positive rate can be bounded by:

$$FP \leq \theta M.$$

To account for this condition, a more restrictive level of significance must be set consistent with the number of unique patterns being evaluated (in our case, the clusters obtained earlier). If c clusters are assessed for goodness, the Bonferroni correction (Bland and Altman, 1995) suggests that the level of significance be set at:

$$\theta' = \frac{\theta}{c}.$$

This addresses the issue of increasing false positives caused by the evaluation of a large number of clusters by correspondingly lowering the p-value required to accept a pattern.

6. Evaluation

All experiments were carried out on a 3.2 GHz P4 with 2 GB of RAM running Linux Fedora Core4. The algorithms to be evaluated were implemented in Java.

6.1 Regulatory Motifs in Drosophila Genome

We evaluated our pattern discovery algorithm on data from the Assessment of Computational Motif Discovery Tools project (Li and Tompa, 2006; Tompa et al., 2005). The focus of this project was to compare 13 motif discovery tools on nucleotide sequences from the human, mouse, fly and yeast genomes to see if they could successfully identify known regulatory elements such as binding sites for genes. These tools differed from each other mainly in their definition of a motif, and in the method used to search for statistically overrepresented motifs. Authors with specific expertise were chosen to test each tool and avoid the disadvantage of being run with an uninformed set of parameters. The expert predictions were then compared with known binding sites in the TRANSFAC database (Wingender et al., 1996) using various statistics to assess the correctness of the predictions.

In our work, we focused on nucleotide sequences from the *Drosophila melanogaster* genome comprising 43 kb base pairs (Li and Tompa, 2006; Tompa et al., 2005). These sequences were divided into 8 data sets, each corresponding to a different binding site. The Drosophila genome was the smallest (in terms of the available data) of the four genomes used in the project, allowing us to compare our method with more computationally intensive algorithms that do not use LSH or clustering. To use our method, we generated random negative sequences with the same background frequencies and lengths as the positive examples, and supplemented both positive and negative sequences with their reverse complements. Our method predicted binding sites in the original data corresponding to all subsequences in the group D_{W_i} with maximum discriminative value. While our algorithm has been designed to use additional information in negative examples when available, this approach presents an example of how our method can be used even in cases where only positive training data is present and negative examples are generated using a simple approach.

The pattern discovery process attempted to find approximate subsequences of lengths 8, 12 and 16. We investigated Hamming radii of 1-3 for patterns of length 8, 2-4 for those of length 12 and 3-5 for length 16. Parameters were chosen using the inequality in Equation 2 so that the LSH probability of false negatives in each case was below 0.005. For each of these cases, we ran our algorithm three times and consistent with the other motif discovery algorithms evaluated, we chose only the best pattern discovered (where best corresponded to lowest rank sum p-value or no result if a pattern with a p-value less than 0.05 was not found).

We compared our method to the 13 motif discovery tools evaluated on the same data set using the *nPC* and *nCC* summary statistics (Li and Tompa, 2006; Tompa et al., 2005). Given the number of nucleotide positions in both known sites and predicted sites (*nTP*), the number of nucleotide positions in neither known sites nor predicted sites (*nTN*), the number of nucleotide positions not in known sites but in predicted sites (*nFP*) and the number of nucleotide positions in known sites but not in predicted sites (*nFN*), the nucleotide level performance coefficient (*nPC*) (Pevzner and Sze, 2000) was defined as:

$$nPC = \frac{nTP}{nTP + nFN + nFP}.$$

The nucleotide level correlation coefficient (*nCC*) (Burset and Guigo, 1996) was defined as:

$$nCC = \frac{nTP.nTN - nFN.nFP}{\sqrt{(nTP+nFN)(nTN+nFP)(nTP+nFP)(nTN+nFN)}}.$$

In addition to comparing our method with results from the 13 motif discovery algorithms evaluated in the Assessment of Computational Motif Discovery Tools project, we also explored the runtime and accuracy of two variations of our algorithm. These variations were meant to assess the contribution of clustering and LSH-based approximate matching to the performance of our method. The first variation we examined did not make use of the clustering process described in Section 4 to reduce overlapping approximate patterns to a smaller group. The second variation further avoided the use of LSH to match subsequences and was based instead on an exhaustive search.

While comparing the three approaches, we use the following notation: *LSHCS* for our original algorithm using clustering and sequential statistics, *NoClust* for the variation that did not use clustering, and *ExhSearch* for the exhaustive search algorithm that further avoided the use of LSH to match subsequences.

For some binding site data sets *NoClust* and *ExhSearch* did not terminate even after very long processing times. We terminated algorithms if they did not produce results within 24 hours of CPU time. These cases are annotated where included.

6.2 Regulatory Motifs in ChIP-on-chip Yeast Data

We evaluated the ability of our method to discover yeast transcription factor binding motifs in ChIP-on-chip data (Harbison et al., 2004). The "gold standard" motifs for this data set were generated by applying a suite of six motif-finding algorithms to intergenic regions from *Saccharomyces cerevisiae* and clustering the results to arrive at a consensus motif for each transcription factor. When no motif was found computationally for the intergenic regions, a literature-based consensus motif was used.

In our experiments, we focused on the 21 transcription factors in the data set for which the six motif-finding algorithms in Harbison et al. (2004) failed to find a significant motif and the reported motif had to be based on a consensus obtained from the literature. For all tests, we used the output from the *LSHCS* algorithm with the lowest rank sum p-value (or no result if a motif with p-value less than 0.05 was not found), with the motif width chosen to match the width of the literature consensus motif. This approach was analogous to earlier work on ChIP-on-chip data to evaluate motif discovery methods (Siddharthan et al., 2005).

For each experiment, we defined positive examples as the set of probe sequences found to bind the transcription factor and set negative examples to be randomly selected non-binding probe sequences. The positive and negative sets contained the same number of sequences. This approach was also chosen to be consistent with earlier studies to evaluate motif discovery tools (Redhead and Bailey, 2007).

6.3 Predictive Morphology Variations in Electrocardiogram Signals

We evaluated our method on 24-hour electrocardiographic (ECG) signals from patients admitted with non-ST-elevation acute coronary syndromes (NSTEACS) to discover patterns of morphology changes associated with future cardiovascular death. Earlier studies suggest that increased beat-to-beat variations in ECG morphology may be associated with instability in the conduction system of the heart and could help identify high risk patients (Syed et al., 2008). We applied our pattern discovery algorithm to discover specific sequences of beat-to-beat changes that had predictive value.

Given a 24-hour ECG signal, we first converted the data recorded during the course of hospitalization into a time-series of morphology changes between pairs of consecutive heart beats (Syed et al., 2008). Morphology changes between beats were measured using a dynamic time-warping algorithm (Rabiner et al., 1978) that calculates the time-aligned energy difference between beats. The morphology change time-series was then converted into a sequence using symbolic aggregate approximation (SAX) (Lin et al., 2003) with an alphabet size of 10. In this manner, multiple ECG signals were transformed into sequences that could be analyzed by our method. On average, each sequence corresponding to 24 hours of ECG was almost 100,000 symbols long.

On a training set of 765 patients, where 15 patients died over a 90 day period following NSTEACS (i.e., 15 positive examples and 750 negative examples), we used our pattern discovery algorithm to learn sequences of morphology changes in the ECG signal that were predictive of death. We searched for patterns of length 8 with a Hamming distance of at most 2. Parameters were chosen using the inequality in Equation 2 so that the LSH probability of false negatives was less than 0.01. We selected patterns that showed a C-statistic greater than 0.7 and a rank sum test p-value of 0.05 corrected for multiple hypotheses using the Bonferroni correction. These were evaluated on a test set of 250 patients with 10 deaths.

We also studied the runtime performance of our algorithm on this data set with and without the use of sequential statistics to find significant patterns. Given the large number of negative examples in the training data, we employed the sequential formulation in Equations 8 and 9 with $\lambda = 0.2$. Consistent with the notation proposed in Section 6 we denote our original algorithm using sequential statistics as *LSHCS* while the variation that avoids sequential statistics is denoted by *NoSeqStats*.

7. Results

The results of our experiments are as follow.

7.1 Regulatory Motifs in Drosophila Genome

Table 1 presents the results of the 13 different motif discovery algorithms evaluated by the Assessment of Computational Motif Discovery Tools project. The results of using our method are given in Table 2.

Algorithm	*nPC*	*nCC*
AlignACE	0	-0.006
ANN-Spec	0.010	0.002
Consensus	0	-0.011
GLAM	0.002	-0.008
Improbizer	0.008	0.002
MEME	0.021	0.027
*MEME*3	0.016	0.013
MITRA	0	-0.008
MotifSampler	0.003	-0.006
Oligodyad-Analysis	0	-0.015
QuickScore	0	-0.016
SeSiMCMC	0.036	0.054
Weeder	0.009	0.011
YMF	0	-0.014

Table 1: Performance of 13 different motif discovery methods on the Drosophila genome (*nPC*=performance coefficient, *nCC*=correlation coefficient) in the Assessment of Computational Motif Discovery Tools project (Tompa et al., 2005).

Parameter	*nPC*	*nCC*
$L = 8, d = 1$	0.021	0.031
$L = 8, d = 2$	0.007	-0.009
$L = 8, d = 3$	0.013	-0.002
$L = 12, d = 2$	0.013	0.018
$L = 12, d = 3$	0.039	0.062
$L = 12, d = 4$	0.013	0.018
$L = 16, d = 3$	0	-0.011
$L = 16, d = 4$	0.032	0.055
$L = 16, d = 5$	0.056	0.093

Table 2: Performance of the *LSHCS* pattern discovery method on the Drosophila genome using different input parameters (*nPC*=performance coefficient, *nCC*=correlation coefficient, *L*=pattern length, *d*=maximum Hamming distance allowed in pattern).

Our *LSHCS* algorithm compared favorably with the other motif discovery algorithms evaluated on the same data by experts. In particular, for two of the parameter settings evaluated (i.e., $L = 12, d = 3$ and $L = 16, d = 5$), our algorithm had both a higher performance coefficient and a higher correlation coefficient than any of the other methods. The $L = 16, d = 4$ case also had a higher correlation coefficient than the motif discovery algorithms previously reported, although the performance coefficient for this choice of parameters was exceeded by *SeSiMCMC*.

We note that these findings help only to validate the ability of our pattern discovery approach to identify potentially interesting activity. Since these results hold on a specific genome, we caution against interpreting the results as a more general comparison of *LSHCS* with nucleotide motif discovery methods. In particular, we observe that the nucleotide motif discovery methods in Table 1 were tested blindly on a single pre-determined choice of parameters. It is possible that with different choices of input parameters (i.e., similar to the evaluation of *LSHCS* using different values of L and d), these methods would have yielded significantly better results.

Tables 3 and 4 present the performance and correlation coefficients for the *NoClust* and *ExhSearch* algorithms. For both these variations, the algorithms did not terminate for the largest choice of Hamming radius d. Investigation revealed that the slow processing times in these cases were associated with insufficient memory to store the neighborhoods for all approximate patterns (i.e., in the absence of clustering for both algorithms). For large choices of the maximum allowed Hamming radius, the neighborhood for each pattern is more extensive and storing this information for all overlapping patterns imposes significant space requirements. Conversely, by using clustering, *LSHCS* is able to reduce the number of overlapping patterns and avoid references to disk.

We found some of the results in Tables 3 and 4 comparing *LSHCS* to *NoClust* and *ExhSearch* to be surprising. In contrast to *LSHCS*, the *NoClust* variation explores all overlapping patterns while *ExhSearch* uses an exhaustive search to find nearest neighbors (i.e., avoiding the small false negative probability associated with LSH). Since both variations use strictly more data and explore the outputs of *LSHCS* as well as other alternatives, we expected the results to improve uniformly between *LSHCS* and *ExhSearch* for all parameter selections. While this was generally the case, for some parameter choices (e.g., $L = 16, d = 4$) the best results were obtained by *LSHCS*. Examining the outputs by all three algorithms revealed these inconsistencies to be the result of resolving ties between patterns with identical rank sum p-values arbitrarily. While presenting the results in this section, we explicitly note this limitation of the objective function (i.e., the rank sum p-value) used for evaluation.

Ignoring parameter choices for which *NoClust* and *ExhSearch* did not terminate, the performance of all three methods was similar as shown in Table 5. A comparison of the running time for all three algorithms is also presented in Table 6. The running time increased significantly both as all overlapping approximate patterns were studied, and when an exhaustive search was used to replace LSH. The increase in runtime due to exhaustive search was considerably more than the effect of examining all overlapping approximate patterns.

7.2 Regulatory Motifs in ChIP-on-chip Yeast Data

The results of applying the *LSHCS* algorithm on the ChIP-on-chip data set for the 21 transcription factors for which the six motif-finding algorithms failed to find a significant motif are shown in Table 7.

Parameter	*nPC*	*nCC*
$L = 8, d = 1$	0.037	0.063
$L = 8, d = 3$†	...	...
$L = 12, d = 2$	0	-0.009
$L = 12, d = 3$	0.032	0.0499
$L = 12, d = 4$†	...	...
$L = 16, d = 3$	0	-0.011
$L = 16, d = 4$	0.020	0.029
$L = 16, d = 5$†	...	...

Table 3: Performance of the *NoClust* pattern discovery method on the Drosophila genome using different input parameters (*nPC*=performance coefficient, *nCC*=correlation coefficient, *L*=pattern length, *d*=maximum Hamming distance allowed in pattern). † Cases where the algorithm did not terminate in 24 hours.

Parameter	*nPC*	*nCC*
$L = 8, d = 1$	0.039	0.065
$L = 8, d = 2$	0.017	0.015
$L = 8, d = 3$†	...	...
$L = 12, d = 2$	0.006	0.005
$L = 12, d = 3$	0.025	0.036
$L = 12, d = 4$†	...	...
$L = 16, d = 3$	0	-0.011
$L = 16, d = 4$	0.009	0.008
$L = 16, d = 5$†	...	...

Table 4: Performance of the *ExhSearch* pattern discovery method on the Drosophila genome using different input parameters (*nPC*=performance coefficient, *nCC*=correlation coefficient, *L*=pattern length, *d*=maximum Hamming distance allowed in pattern). † Cases where the algorithm did not terminate in 24 hours.

Parameter	Average *nPC*	Average *nCC*
LSHCS	0.019	0.024
NoClust	0.021	0.031
ExhSearch	0.016	0.020

Table 5: Comparison of *LSHCS*, *NoClust* and *ExhSearch* by summarizing results from parameter selections where all three algorithms terminated within 24 hours (*nPC*=performance coefficient, *nCC* =correlation coefficient, *L*=pattern length, *d*=maximum Hamming distance allowed in pattern).

Parameter	*LSHCS* Time	*NoClust* Time	*ExhSearch* Time
$L = 8, d = 1$	5:01	15:40	3:01:22
$L = 8, d = 2$	5:53	1:06:06	3:49:08
$L = 8, d = 3$†	4:13	…	…
$L = 12, d = 2$	18:14	35:54	17:05:37
$L = 12, d = 3$	24:48	2:33:12	17:27:43
$L = 12, d = 4$†	27:30	…	…
$L = 16, d = 3$	31:08	29:59	18:21:15
$L = 16, d = 4$	32:20	2:59:14	18:04:59
$L = 16, d = 5$†	24:15	…	…

Table 6: Time taken for *LSHCS*, *NoClust* and *ExhSearch* pattern discovery methods on the Drosophila genome using different input parameters. † Cases where one or more of the algorithm did not terminate in 24 hours.

Transcription Factor	Motif Found
ADR1	…
DAL80	Yes
GAL80	…
GCR1	…
GZF3	…
HAP2	…
HAP3	Yes
HAP5	…
MAC1	Yes
MET31	…
MET32	…
MOT3	Yes
MSN4	…
PUT3	…
RGT1	…
RLM1	…
ROX1	Yes
RTG3	…
SKO1	…
YAP6	Yes
YOX1	…

Table 7: Results of *LSHCS* on the ChIP-on-chip data set for transcription factors (TF) where common motif-finding algorithms failed to find a significant motif.

Pattern (Centroid)	Rank Sum P-Value	C-statistic
ABCCDFGJ	0.025	0.71
FFJJJJCC	0.004	0.70

Table 8: Statistical significance of approximate patterns found on a training set of 765 post-NSTEACS patients (15 deaths over a 90 day follow-up period) when evaluated on a test population of 250 patients (10 deaths).

In 6 of the 21 experiments conducted, the discriminative motif with the lowest p-value found by the *LSHCS* algorithm corresponded to the consensus motif in the literature, but was not found by any of the six motif-finding algorithms evaluated earlier. *LSHCS*, as well as the other six motif-finding algorithms, failed to find the discriminative motif in the remaining 15 cases although motifs are described in the literature.

7.3 Predictive Morphology Variations in Electrocardiogram Signals

Our pattern discovery method returned 2 approximate patterns that were assessed to have discriminative value in the training set (i.e., a C-statistic of more than 0.7 and a p-value of less than 0.05 after accounting for the Bonferroni correction). Representing the symbols obtained using SAX by the letters *A*-*J*, where *A* corresponds to the symbol class for the least beat-to-beat change in morphology and *J* denotes the symbol for the greatest change, the centroids for the approximate pattern can be written as:

$$ABCCDFGJ$$

$$FFJJJJCC$$

The first of these patterns is equivalent to increasing time-aligned energy changes between successive beats. This may suggest increased instability in the conduction system of the heart. The second pattern corresponds to a run of instability followed by a return to baseline. This pattern can be interpreted as a potential arrhythmia.

The results of testing both patterns on previously unseen data from 250 patients (with 10 deaths over a 90 day follow-up period) are shown in Table 8. Both patterns found by our approximate pattern discovery algorithm showed statistical significance in predicting death according to the C-statistic and rank sum criteria.

A comparison of the running times for the *LSHCS* and *NoSeqStats* algorithms is presented in Table 9. While the outputs produced by both algorithms were identical, the use of sequential statistics helped our *LSHCS* method decrease the runtime of the search process to almost half of what we encountered for the variation not using the methods proposed in Section 5.3. We also note that the *NoSeqStats* variation used considerably more memory than the *LSHCS* approach. This effect was due to *LSHCS* purging state for patterns that did not obey the inequality in Equation 9. In the absence of sequential statistics, *NoSeqStats* had to retain ranking information for all patterns till the training data set was completely analyzed.

Algorithm	Time
LSHCS	5:08:24
NoSeqStats	9:43:09

Table 9: Time taken by the *LSHCS* and *NoSeqStats* pattern discovery algorithms on the cardiovascular training data set.

8. Summary and Discussion

This paper presents an approach to efficiently learn patterns in labeled sequences that can be used for classification. We define patterns as groups of approximately matching subsequences, that is, all variations of a subsequence within a given Hamming radius. Our method represents a fully automated approach for unsupervised pattern discovery, where the goodness of patterns is assessed by measuring differences in their frequency of occurrence in positive and negative training examples.

We designed our pattern discovery algorithm to make it both accurate and efficient in terms of space and computation. We briefly review the central ideas of our work.

First, we include patterns from both positive and negative training sets in our search for discriminative activity. In many applications, we can consider positive examples as being associated with some physical phenomenon. Patterns that occur mainly in positive examples can be considered as potentially regulatory activity that may cause the phenomenon being studied. Conversely, patterns that occur mainly in negative examples (i.e., are absent in positive examples) can be considered as potentially supressive activity. The absence of these suppressive patterns in positive examples may promote the observed phenomenon. We allow for the discovery of both types of activity. This approach has a further advantage in that it reduces the need for prior knowledge. In the absence of negative examples, activity in positive training samples must be assessed through some assumption of statistical over-representation. By being able to directly compare positive examples against negative ones, we attempt to remove the need for such assumptions.

Second, to efficiently search for approximate patterns, we make use of locality sensitive hashing. LSH has been proposed as a randomized approximation algorithm to solve the nearest neighbor problem. We employ an iterative LSH method that is able to efficiently find groups of matching subsequences for subsequent analysis as candidate patterns.

Third, we explore the idea of clustering together subsequences, so that the number of candidate patterns can be reduced. This abstraction is intended to decrease the redundancy associated with evaluating a large number of approximate patterns with significant overlap. Reducing this redundancy, while still spanning the search space, provides an improvement in efficiency and also allows for more clarity in interpreting the results of the search process.

Finally, we make use of non-parametric statistical methods that have been designed to identify patterns with different distributions in both positive and negative examples. An extension of this work is the use of sequential methods, which only use as much of the training data as is needed to make a decision about a candidate pattern.

We evaluated the use of our pattern discovery algorithm on data from different real-world applications. On nucleotide sequences from the Drosophila genome, our method was able to discover approximate binding sites that were preserved upstream of genes. A similar result was seen for ChIP-on-chip data from the yeast genome. For cardiovascular data from patients admitted with acute coronary syndromes, our pattern discovery approach identified approximately conserved sequences of morphology changes that were predictive of future death in a test population. Our data showed that the use of LSH, clustering, and sequential statistics improved the running time of the search algorithm by an order of magnitude without any noticeable effect on accuracy. These results suggest that our methods may allow for an efficient unsupervised approach to learn interesting dissimilarities between positive and negative examples, which may have a functional role.

Acknowledgments

We would like to thank Dorothy Curtis for helping with the computational needs for this project, Murtaza Zafar and Ebad Ahmed for their technical input regarding sequential statistics, and Manolis Kellis for suggesting the genomic data set for validation. We also thank the editors and anonymous reviewers for their feedback.

This work was supported by the Center for Integration of Medicine and Innovative Technology (CIMIT), the Harvard-MIT Division of Health Sciences and Technology (HST), the Industrial Technology Research Institute (ITRI), and Texas Instruments (TI).

References

Timothy L. Bailey and Charles Elkan. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In *International Conference on Intelligent Systems in Molecular Biology*, pages 28–36, 1994.

Yosef Barash, Gill Bejerano, and Nir Friedman. A simple hyper-geometric approach for discovering putative transcription factor binding sites. *Algorithms in Bioinformatics*, pages 278–293, 2001.

Serafim Batzoglou, Lior Pachter, Jill P. Mesirov, Bonnie Berger, and Eric S. Lander. Human and mouse gene structure: comparative analysis and its application to exon prediction. *Genome Research*, pages 950–958, 2000.

Martin Bland and Douglas G. Altman. Multiple significance tests: the bonferroni method. *British Medical Journal*, page 170, 1995.

Alvis Brazma, Inge Jonassen, Ingvar Eidhammer, and David Gilber. Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, pages 279–305, 1998.

Jeremy Buhler. Efficient large-scale sequence comparison by locality-sensitive hashing. *Bioinformatics*, pages 419–428, 2001.

Jeremy Buhler and Martin Tompa. Finding motifs using random projections. *Journal of Computational Biology*, pages 225–242, 2002.

Moises Burset and Roderic Guigo. Evaluation of gene structure prediction programs. *Genomics*, pages 353–367, 1996.

Stuart Daw, Charles E. Finney, and Eugene R. Tracy. A review of symbolic analysis of experimental data. *Review of Scientific Instruments*, pages 915–930, 2003.

Arthur L. Delcher, Simon Kasif, Robert D. Fleischmann, Jeremy Peterson, Owen White, and Steven L. Salzberg. Alignment of whole genomes. *Nucleic Acids Research*, pages 2369–2376, 1999.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

Eleazar Eskin and Pavel Pevzner. Finding composite regulatory patterns in dna sequences. *Bioinformatics*, pages 354–363, 2002.

Jean D. Gibbons. *Nonparametric Statistical Inference*. Marcel Dekker, 1985.

William N. Grundy, Timothy L. Bailey, Charles P. Elkan, and Michael E. Baker. Meta-meme: motif-based hidden markov models of protein families. *Computer Applications in the Biosciences*, pages 397–406, 1997.

Jaiwei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kauffman, 2005.

James A. Hanley and Barbara J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, pages 29–36, 1982.

Christopher T. Harbison, Benjamin Gordon, Tong I. Lee, Nicola J. Rinaldi, Kenzie D. Macisaac, Timothy W. Danford, Nancy M. Hannett, Jean-Bosco Tange, David B. Reynoalds, Jane Yoo, Ezra G. Jennings, Julia Zeitlingen, Dmitry K. Pokholok, Manolis Kellis, Alex Rolfe, Ken T. Takusagawa, Eric S. Lander, David K. Gifford, Ernest Fraenkel, and Richard A. Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, pages 99–104, 2004.

Taher H. Haveliwala, Aristides Gionis, and Piotr Indyk. Scalable techniques for clustering the web. In *WebDB Workshop*, 2000.

Dorit S. Hochbaum and David B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, pages 180–184, 1985.

Myles Hollander and Douglas A. Wolfe. *Nonparametric Statistical Methods*. John Wiley and Sons, 1999.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Symposium on Theory of Computing*, pages 604–613, 1998.

Tommi Jaakkola, Mark Diekhans, and David Haussler. Using the fisher kernel method to detect remote protein homologies. In *Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 149–158, 1999.

Manolis Kellis, Nick Patterson, Bruce Birren, Bonnie Berger, and Eric S. Lander. Methods in comparative genomics: genome correspondence, gene identification and regulatory motif discovery. *Journal of Computational Biology*, pages 319–355, 2004.

Anders Krogh. Hidden markov models for labeled sequences. In *IEEE International Conference on Pattern Recognition*, pages 140–144, 1994.

Charles E. Lawrence, Stephen F. Altschul, Mark S. Boguski, Jun S. Liu, Andrew F. Neuwald, and John C. Wootton. Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment. *Science*, pages 208–214, 1993.

Erich L. Lehmann. *Nonparametrics: Statistical Methods Based on Ranks*. McGraw-Hill, 1975.

Christina Leslie, Eleazar Eskin, Jason Weston, and William S. Noble. Mismatch string kernels for svm protein classification. *Advances in Neural Information Processing Systems*, 2003.

Henry Leung and Francis Chin. Finding motifs from all sequences with and without binding sites. *Bioinformatics*, pages 2217–2223, 2006.

Jiuyong Li, Ada W. Fu, Hongxing He, Jie Chen, Huidong Jin, Damien McAullay, Graham Williams, Ross Sparks, and Chris Kelman. Mining risk patterns in medical data. In *ACM Conference on Knowledge Discovery in Data*, pages 770–775, 2005.

Nan Li and Martin Tompa. Analysis of computational approaches for motif discovery. *Algorithms for Molecular Biology*, page 8, 2006.

Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003.

Shirley Liu, Douglas L Brutlag, and Jun S. Liu. Bioprospector: discovering conserved dna motifs in upstream regulatory regions of co-expressed genes. In *Pacific Symposium on Biocomputing*, pages 127–138, 2001.

Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *Very Large Data Bases*, pages 950–961, 2007.

Bamshad Mobasher, Namit Jain, Eui-Hong Han, and Jaideep Srivasta. Web mining: pattern discovery from world wide web transactions. *Department of Computer Science, University of Minnesota Technical Report TR-96-050*, 1996.

Giulio Pavesi, Giancarlo Mauri, and Graziano Pesole. An algorithm for finding signals of unknown length in dna sequences. *Bioinformatics*, pages 207–214, 2001.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

Pavel A. Pevzner and Sing-Hoi Sze. Combinatorial approaches to finding subtle signal in dna sequences. In *International Conference on Intelligent Systems for Molecular Biology*, pages 269–278, 2000.

Ravi M. Phatarfod and Aidan Sudbury. A simple sequential wilcoxon test. *Australian Journal of Statistics*, pages 93–106, 1988.

Lawrence R. Rabiner, Aaron E. Rosenberg, and Stephen E. Levinson. Considerations in dynamic time warping algorithms for discrete word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 575–582, 1978.

Emma Redhead and Timothy L. Bailey. Discriminative motif discovery in dna and protein sequences using the deme algorithm. *BMC Bioinformatics*, page 385, 2007.

Marion R. Reynolds. A sequential signed-rank test for symmetry. *The Annals of Statistics*, pages 382–400, 1975.

Michael J. Shaw, Chandrasekar Subramaniam, Gek W. Tan, and Michael E. Welge. Knowledge management and data mining for marketing. *Decision Support Systems*, pages 127–137, 2001.

Rahul Siddharthan, Eric D. Siggia, and Erik van Nimwegen. Phylogibbs: a gibbs sampling motif finder that incorporates phylogeny. *PLoS Computational Biology*, pages 534–556, 2005.

Saurabh Sinha and Martin Tompa. Ymf: a program for discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic Acids Research*, pages 3586–3588, 2003.

Zeeshan Syed, John V. Guttag, and Collin M. Stultz. Clustering and symbolic analysis in cardiovascular signals: discovery and visualization of medically relevant patterns in long-term data using limited prior knowledge. *EURASIP Journal on Advances in Signal Processing*, 2007.

Zeeshan Syed, Benjamin M. Scirica, Collin M. Stultz, and John V. Guttag. Risk-stratification following acute coronary syndromes using a novel electrocardiographic technique to measure variability in morphology. In *Computers in Cardiology*, 2008.

Saeed Tavazoie, Jason D. Hughes, Michael J. Campbell, Raymond J. Cho, and George M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, pages 281–285, 1999.

Martin Tompa, Nan Li, Timothy L. Bailey, George M. Church, Bart D. Moor, Eleazer Eskin, Alexander V. Favorov, Martin C. Frith, Yutao Fu, James Kent, Vsevolod J. Makeev, Andrei A. Mironov, William S. Noble, Giulio Pavesi, Graziano Pesole, Mireille Regnier, Nicolas Simonis, Saurabh Sinha, Gert Thijs, Jacques V. Helden, Mathias Vandenbogaert, Zhiping Weng, Christopher Workman, Chun Ye, and Zhou Zhu. Assessing computational tools for the discovery of transcription factor binding sites. *Nature Biotechnology*, pages 137–144, 2005.

Jason T.L. Wang, Bruce A. Shapiro, and Dennis E. Shasha. *Pattern Discovery in Biomolecular Data: Tools, Techniques, and Applications*. Oxford University Press, 1999.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, pages 80–83, 1945.

Edgar Wingender, Peter Dietze, Holger Karas, and Rainer Knuppel. Transfac: a database on transcription factors and their dna binding sites. *Nucleic Acids Research*, pages 238–241, 1996.