Combining Text Mining and Sequence Analysis to Discover Protein Functional Regions

E. Eskin and E. Agichtein

Pacific Symposium on Biocomputing 9:288-299(2004)

COMBINING TEXT MINING AND SEQUENCE ANALYSIS TO DISCOVER PROTEIN FUNCTIONAL REGIONS

E. ESKIN
School of Computer Science Engineering
Hebrew University
eeskin@cs.huji.ac.il

E. AGICHTEIN
Department of Computer Science
Columbia University
eugene@cs.columbia.edu

Recently presented protein sequence classification models can identify relevant regions of the sequence. This observation has many potential applications to detecting functional regions of proteins. However, identifying such sequence regions automatically is difficult in practice, as relatively few types of information have enough annotated sequences to perform this analysis. Our approach addresses this data scarcity problem by combining text and sequence analysis. First, we train a text classifier over the explicit textual annotations available for some of the sequences in the dataset, and use the trained classifier to predict the class for the rest of the unlabeled sequences. We then train a joint sequence text classifier over the text contained in the functional annotations of the sequences, and the actual sequences in this larger, automatically extended dataset. Finally, we project the classifier onto the original sequences to determine the relevant regions of the sequences. We demonstrate the effectiveness of our approach by predicting protein sub-cellular localization and determining localization specific functional regions of these proteins.

1 Introduction

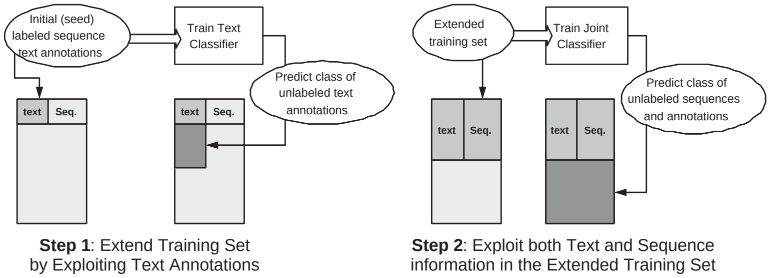
Supervised learning techniques over sequences have had a tremendous amount of success in modeling proteins. Some of the most widely used methods are Hidden Markov Models (HMMs) to model protein families [1,2,3] and neural network techniques for predicting secondary structure [4].

Recently a new class of models which use *margin* learning algorithms such as the Support Vector Machine (SVM) algorithm, have been applied to modeling protein families. These models include the spectrum kernel[5] and mismatch kernel[6] which have been shown to be competitive with state-of-the-art methods for protein family classification. These methods represent sequences as collections of short substrings of length k or k-mers. One property of these classifiers is that we can examine the trained models generated by these methods and discover which k-mers are the most important for discriminating between the classes. By projecting these k-mers onto the original sequences, we can discover which regions of the protein specifically correspond to the class and potentially discover the relevant functional region of the protein. In a recent paper, it has been shown that some of the k-mers with the highest weights in a protein family classification model correspond to known motifs of the protein family [7].

This technique is general in that it can be applied to determine the relevant functional region of a set of proteins given a set of example proteins by creating a data set where the examples of the class of proteins are positive training examples and a sampling of other proteins are negative examples. However, despite the large size of protein databases and the large amount of annotated proteins, very few types of information are sufficiently annotated to generate a large enough training set of proteins to perform this analysis. For example, consider the sub-cellular localization of proteins. Only a very small fraction of the database, 15%, is annotated with sub-cellular localization despite the fact that 35% of the database is annotated with functional annotation which corresponds to localization. If we can somehow use the functional annotation as a proxy for localization information, we can then apply our analysis to identify the regions of the proteins that are specific to each sub-cellular location. In their recent work, Nair and Rost [8], defined a method for inferring localization information from the functional annotation which greatly influenced our work.

In this paper, we introduce a framework that combines text-mining over database annotations with sequence learning to both classify proteins and determine the functional regions specific to the classes. Our framework is designed specifically for the case when we are given a relatively small set of example sequences compared to a much larger amount of text annotated, yet unlabeled sequences. Our framework learns how the text is correlated with the labels and jointly learns over sequences and text of both the example (labeled) and unlabeled (yet annotated) examples. The output of the learning is a sequence classifier which can be used to identify the regions in the proteins specific to the class.

We demonstrate our method with a proof of concept application to identify regions correlated to sub-cellular localization. We choose sub-cellular location as the proof of concept application because two recent works by Nair and Rost [8,9] show that functional annotations of proteins correlate with localization and localization can be inferred from sequences. Using the small set of labeled examples and sequences as a seed we train a text classifier to predict the sub-cellular localization based on the functional annotations similar to the approach presented in Nair and Rost, 2002[8]. This effectively augments the seed set of labeled sequences with a larger set of sequences with predicted localizations. We then jointly learn a sequence and text classifier over the extended dataset.


Initial (seed) labeled sequence text annotations
Train Text Classifier
Predict class of unlabeled text annotations
text
Seq.
text
Seq.
Extended training set
Train Joint Classifier
Predict class of unlabeled sequences and annotations
text
Seq.
text
Seq.

Step 1: Extend Training Set by Exploiting Text Annotations

Step 2: Exploit both Text and Sequence information in the Extended Training Set

Figure 1: Framework for Extending and Combining Textual Annotations with Sequence Data.

This is similar to the work by Nair and Rost, 2002[9] where they showed that sequence homology can be used to predict localization. Finally we then use the sequence model to identify the localization specific regions of the proteins. Preliminary analysis of the regions shows that some correspond with known biological sites such as DNA-binding sites for the nuclear proteins.

2 Methods

2.1 Framework Overview

The framework for discovering functional regions of a proteins given a set of examples of the protein consists of several steps, as shown in Figure 1. First we create a seed dataset which consists of the labeled proteins as positive training examples and a sampling of other proteins as the negative examples. Using this seed set, we train a text classifier over the annotations of the sequences. Then using the text classifier, we predict over the database additional sequences which correspond to the class. Using this extended dataset, we train a joint sequence and text classifier. By projecting the classifier onto our original sequences, we can identify which regions of the protein have a high positive weight with respect to the class corresponding to the example proteins and are likely candidates for the relevant functional region of the protein.

The input to our framework is a set of examples of the proteins and the output is a joint text sequence classifier for predicting other examples of that protein and predictions for regions in the original proteins which correspond to the common function of the example set of proteins.

2.2 Extending the Seed Dataset

A significant problem in machine learning is the scarcity of training data. Insufficient training data often prevents machine learning techniques from achieving acceptable accuracy. In this section we present an application of text classification that allows us to *automatically* construct a comprehensive training set by starting with the initial smaller *seed* set of labeled sequences. Combining labeled and unlabeled examples is a topic that has been thoroughly studied in the machine learning community (e.g., Blum and Mitchell, 1998[10] and Tsuda *et. al.* ,[11] and the references therein for a starting point). The simple approach that we describe below was sufficient for our task, and we plan to explore more sophisticated approaches in our future work.

To extend the training set, we exploit the large amount of textual information often associated with a sequence. For example, SWISS-PROT[12] provides rich textual annotations for each entry in the database. Unfortunately, these annotations are difficult to compile and maintain, and as a result important information is often missing for many entries (e.g., the localization information). However, we can sometimes deduce this missing information from the textual annotations that happen to be present for a database entry. This general approach was presented in Nair and Rost[8].

The predictions for the unknown sequences rely on some form of *classifying* the textual annotations. After training over a number of labeled training examples, text classifiers can successfully predict the correct class of unlabeled texts. We represent the text using a *bag of words* model where each text annotation is mapped to a vector containing the frequency of each word. As the actual classifier, we use RIPPER[13], a state of the art text classification system. RIPPER operates by learning *rules* to describe the text in the training examples, and then applies these rules to predict the appropriate classification of new, unlabeled texts.

2.3 Training a Joint Sequence Text Classifier

Each protein record consists of the sequence and the text from its functional annotation. We construct a classifier to predict members of the class of proteins corresponding to the example proteins by learning from both the text and the sequences. In order to learn from the text and sequences jointly we use a kernel framework. Both sequences and text are mapped to points in a *feature space* which is a high dimensional vector space. A kernel for both sequences and text allows us to efficiently compute inner products between points in the space. Using this kernel, we apply the SVM algorithm to train.

The kernel, described below, is constructed in such a way to take into account interactions between the text and sequences during the learning, which results in a true joint sequence text classifier.

Text Kernel

The feature space for the text portion of a protein record uses the bag of words representation described above. The feature space corresponding to the kernel is a very high dimensional space where each dimension corresponds to a specific word. Each word w corresponds to a vector in the space, $\phi_T(w)$ where the value of the vector 1 for the word's dimension and 0 for all other dimensions. A text string x is mapped to a vector which is the sum of the vectors corresponding to the words in the text, $\phi_T(x) = \sum_{w \in x} \phi_T(w)$.

Although the dimension of the feature space is very large, the vectors corresponding to the text strings are very sparse. We can take advantage of this to compute inner products between points in the feature space very efficiently. For two text annotations x and y, we denote the text kernel to be $K_T(x, y) = \phi_T(x) \cdot \phi_T(y)$.

Sequence Kernel

Sequences are also represented as points in a high dimensional feature space. Sequences are represented as a collection of their substrings of a fixed length k (or k-mers) obtained by sliding a window of length k across the length of the sequence. The simplest sequence feature space contains a dimension for each possible k-mer for a total dimension of 20^k. For a k-mer a, the image of the k-mer in the sequence feature space, $\phi_S(a)$, has the value 1 for the k-mer a and the value 0 for the other dimensions. The image of a sequence x is the sum of the images of its k-mers, $\phi_S(x) = \sum_{a \in x} \phi_S(a)$. This sequence representation is equivalent to the k-spectrum kernel[5]. An advantage of this representation is that we can compute kernels or inner products of points in the feature space very efficiently using a trie data structure[5].

In practice, because of mutations in the sequences, exact matching k-mers between sequences are very rare. In order to more effectively model biological sequences, we use the *sparse kernel* sequence representation that allows for approximate matching. The sparse kernel is similar in flavor to the mismatch kernel and is fully described elsewhere[14,15].

Consider two sequences of length k, a and b. Each sequence consists of a single substring. The sparse kernel defines a mapping into a feature space which has the following property

$$\phi(a) \cdot \phi(b) = \alpha^{d_H(a,b)} \tag{1}$$

where $d_H(a,b)$ is the Hamming distance between substrings a and b and $0 < \alpha < 1$ is a parameter in the construction of the mapping. If the two substrings are identical, than the Hamming distance is zero and the substrings contribute 1 to the inner product of the sequences, exactly as in the spectrum kernel. However, if the Hamming distance is greater than zero, the similarity is reduced by a factor of α for every mismatch. Details of the sparse kernel implementation are described elsewhere[14,15].

Combining Text and Sequences

We can use the framework of kernels to define a feature space which allows for interactions between sequences and text annotations. In our approach, we use a very simple method for combining the text and sequence classifiers. There exists a vast literature in machine learning on alternative techniques for this problem.

We now define our combined kernel $K_C(x,y) = K_T(x,y) + K_S(x,y) + (K_T(x,y) + K_S(x,y))^2$. The first two terms effectively include the two feature spaces of text and sequences. The third term is a degree two polynomial kernel over the sum of the two kernels. If we explicitly determine the feature map for the combined kernel, the third term would include features for all pairs of sequences and words. Since the classifier trains over this space, it effectively learns from both sequence and text and the interactions between them.

Support Vector Machines

Support Vector Machines (SVMs) are a type of supervised learning algorithms first introduced by by Vapnik[16]. Given a set of labeled training vectors (positive and negative input examples), SVMs learn a linear decision boundary to discriminate between the two classes. The result is a linear classification rule that can be used to classify new test examples.

Suppose our training set consists of labeled input vectors $(\mathbf{x_i}, \mathbf{y_i})$, $i = 1 \ldots m$, where $\mathbf{x_i} \in \mathbb{R}^\mathbf{n}$ and $y_i \in \{\pm 1\}$. We can specify a linear classification rule f by a pair $(\mathbf{w}, b)$, where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$, via

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + \mathbf{b}, \tag{2}$$

where a point $\mathbf{x}$ is classified as positive (negative) if $f(\mathbf{x}) > \mathbf{0}$ ($f(\mathbf{x}) < \mathbf{0}$). Such a classification rule corresponds to a linear (hyperplane) decision boundary between positive and negative points. The SVM algorithm computes a hyperplane that satisfies a trade-off between maximizing the geometric margin which is the distance between positive and negative labeled points and training errors. A key feature of any SVM optimization problem is that it is equivalent to solving a dual quadratic programming problem that depends only on the inner products $\mathbf{x_i} \cdot \mathbf{x_j}$ of the training vectors which allows for the application of kernel techniques. For example, by replacing $\mathbf{x_i} \cdot \mathbf{x_j}$ by $K_C(x_i, x_j)$ in the dual problem, we can use SVMs in our combined text sequence feature space.

2.4 Predicting Relevant Functional Regions

Once a SVM is trained over a set of data, the classifier is represented in its dual form as a set of support vector weights s_i, one for each training example x_i. The form of the SVM classifier is

$$f(x) = \sum_i s_i K(x, x_i) = \sum_i s_i \phi(x) \cdot \phi(x_i) \tag{3}$$

which can be represented in the primal form as

$$f(x) = \phi(x) \cdot \sum_i s_i \phi(x_i) = \phi(x) \cdot \mathbf{w} \tag{4}$$

where $\mathbf{w} = \sum_\mathbf{i} \mathbf{s_i} \phi(\mathbf{x_i})$ is the SVM hyperplane.

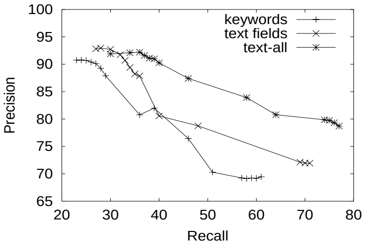
By explicitly computing $\phi(x_i)$ we can compute $\mathbf{w}$ directly. In the case of sequences, this can be efficiently implemented using the same data structures used for computing kernels between sequences[14,15]. We are interested in the sequence only portion of the feature space. For the sequence portion, $\mathbf{w}$ has a weight for every possible k-mer. The score can be interpreted as a measure of how discriminative the k-mer is with respect to the classifier. High positive scores correspond to k-mers that tend to occur in the example set and not in other proteins. We define the score for a region on the protein as the sum of the k-mer scores contained in the region. If a region score is above a threshold, we predict that the region is a potential functional region associated with the example proteins.

3 Results for Protein Localization

We evaluate our framework in three ways. First we measure the accuracy of extending the set of labeled examples. Second, we evaluate the joint text sequence classifier over 20% of the *annotated* localization data. This data was held out of the training in all steps of the framework. We evaluate the accuracy of predicting localization from the functional class over this data. We also evaluate the joint sequence text classifier over this data and compare it to a text only and sequence only classifier. Finally, we perform a preliminary

Localization	*Annotated Count*	*Predicted Count*	*Precision*	*Recall*
cytoplasm	4,976	28,318	0.869	0.705
nuclear	3,843	10,504	0.949	0.790
mitoch	1,925	6,996	0.823	0.656
chloroplast	1,693	3,414	0.869	0.705
extracel	755	7,724	0.728	0.474
endoplas	655	2,742	0.696	0.538
perox	174	810	0.442	0.217
golgi	169	914	0.805	0.559
lyso	167	1,004	0.654	0.530
vacuolar	97	470	0.579	0.112
Total	14,454	62,896		

(a)



(b)

Figure 2: **(a):** Explicitly Annotated localization, and localization predicted based on textual information for SWISS-PROT4.0., **(b):** Precision vs. recall of the text classifier using *keywords* only, vs. field-specific text annotations, vs. *all* available text annotations

analysis of the predictions of the relevant regions in the proteins to localization. We specifically examine nuclear localization signals since many of these are well known and there are readily available databases which we can use to verify our predictions.

3.1 Data Description

We use SWISS-PROT4.0 [12], a large database of sequences and associated textual annotations. In this proof of concept application, we focus on the specific task of inferring sub-cellular localization. A fraction of the sequences in SWISS-PROT have associated annotations that explicitly state their subcellular localization. We report the number of sequences with explicitly annotated localization of each type in Table 2.

As we can see, out of more than 100,000 entries in SWISS-PROT, less than 15% have explicit localization information.

3.2 Increasing the Set of Localization Annotated Sequences

We can increase the amount of information available to a learner by *augmenting* the explicitly labeled examples with unlabeled data. Useful information relevant to localization is often contained within unlabeled text annotations. By learning to recognize the textual annotations associated with localizations, we can assign localization labels to the unlabeled text annotated sequences.

This general approach for predicting localization of unlabeled, but annotated, sequences is presented in Nair and Rost [8]. In their approach, the training focuses on detecting a set of *discriminating* keywords. If such a keyword is present, the sequence is predicted to belong to the appropriate class. In this work we used RIPPER[13], a rule-based classifier, to learn *rules* to predict localization of an SWISS-PROT entry based on textual annotations. The classifier was trained over the 14,454 explicitly annotated sequences. The derived rules were then used to *predict* the localization of the remaining (unlabeled) SWISS-PROT entries.

The approach described in Nair and Rost, 2002[8] focuses on carefully selected and assigned *keyword* annotations, and does not consider the *unstructured* annotations that are often available for the sequences. Text classification systems such as RIPPER implement sophisticated feature selection algorithms, and can be trained over the noisy, but potentially informative *unstructured* data. To evaluate this hypothesis, we varied the types of textual annotations available to the classifier. We compared the quality of prediction based only on the *keywords* information, as used in Nair and Rost[8], to the prediction accuracy achieved by considering other text fields, such as *descriptions*, and finally with using *all* of the available textual annotations for the sequence.

The experimental results for varying the type of textual annotation are reported in Figure 2(b). While the specific evaluation setup and methodology that we used is slightly different from the evaluation of Nair and Rost[8] for the same task, the overall results for *keywords*-based classification appear comparable. As we can see in Figure 2(b), considering *all* of the available textual annotations significantly increases both the recall and the precision of predicting the localization of unknown sequences. For example, at the precision level of 80%, using all of the text annotations allows RIPPER to achieve significantly higher recall. Therefore, for the remainder of this paper our text classifier considers *all* of the textual annotations that are available for each SWISS-PROT entry.

The counts of the automatically predicted SWISS-PROT entries are reported in Figure 2(a). We also report the precision and recall of the classifier, evaluated over the hold-out data using cross-validation. These accuracy figures serve as an estimate of the accuracy, or the "quality" of the resulting extended training set. Note that the while the text classifier introduces some noise into the training set, the extended training set at over 62,000 examples is significantly larger than the original training set. This extended *automatically* labeled training set can now be used to train a better join text and sequence classifier.

3.3 Evaluation the Joint Text Sequence Classifier

Over the extended data described in Section 3.2, we performed experiments to measure the improvement of the classifier when considering text and sequences

Localization Category	*Text Classifier*	*Sequence Classifier*	*Joint Classifier*
cytoplasm	0.91	0.86	0.93
nuclear	0.94	0.91	0.97
mitoch	0.96	0.91	0.99
chloroplast	0.96	0.96	0.96
extracel	0.92	0.93	0.95
endoplas	0.89	0.94	0.96
perox	0.93	0.88	0.95
golgi	0.91	0.83	0.93
lyso	0.93	0.99	0.99
vacuolar	0.94	0.94	0.94

Table 1: Comparison of text only classifier, sequence only classifier and joint classifier for each localization category. Each classifier is evaluated by computing the ROC_{50} score.

together. We ran three experiments by leaving out 20% of the original annotated sequence data as a test set and using the remaining data as a training set. We trained three models on the training set: a text only classifier, a sequence only classifier and a joint sequence text classifier. For all three classifiers, we used the SVM algorithm with the only difference being the choice of kernel. The text classifier uses the text kernel $K_T(x, y)$, the sequence classifier uses the sequence kernel $K_S(x, y)$ and the combined classifier use the $K_C(x, y)$ kernel. For each class, we used all of the members of the class as positive examples and a sampling of the remaining classes as negative examples. For each of the classes of localization data we report the results of the classifiers performance over the test data in Table 2(b).

We use ROC_{50} scores to compare the performance of different homology detection between methods. The ROC_{50} score is the area under the receiver operating characteristic curve — the plot of true positives as a function of false positives — up to the first 50 false positives[17]. A score of 1 indicates perfect separation of positives from negatives, whereas a score of 0 indicates that none of the top 50 sequences (or text annotations) selected by the algorithm were positives.

3.4 Identifying Regions Relevant to Localization

We made predictions for regions correlated to localization using the method described in Section 2.4. Since of all the localization signals, nuclear localization signals are the most characterized and have a searchable database of signals, the NLS database[18], we restricted our evaluation to these signals. We examined the 20 highest non-overlapping regions and compared them to the NLS database and found 8 common signals. Table 2 shows the eight predicted regions and the corresponding entries from the NLS database.

Predicted Region	NLS Signal	 Origin	 Reference
KKKKKKK	KKKKKx3,6KK	predicted	–
RKRKK	RKRKK	experimental	(A)
KKEKKEKKDKKEKKEKKEKKDKKEKKEKKEKK	KKEKKKSKK	experimental	(B)
GGGTGGTGTGTGGG	RGGRGRGRG	predicted	–
QRFTQRGGGAVGKNRRGGRGGNRGGRNNNSTR	GGGxxxKNRRxxxxxxRGGRN	experimental	(C)
EVLKVQKRRIYD	[PL]KxxKRR	predicted	–
LSGGTPKRCLDLSNLS	T[PLV]KRC	predicted	–

Table 2: Eight predicted regions corresponding to nuclear localization and the corresponding entries from the NLS database. The signal entry is a signal that is close to the predicted signal. The origin describes whether the signal was experimentally verified or predicted according to the database and the reference is the corresponding reference for the predicted signals. References: (A) Bouvier, D., Badacci, G., Mol, Biol. Cell,1995,6,1697-705 (B) Youssoufian, H. et al., Blood Cells Mol. Dis.,1999,25,305-9. (C) Truant, R., Cullen, B.R., Mol. Cell. Biol., 1998,18,1449-1458.

4 Discussion

We have presented a framework for combining textual annotations and sequence data for determining the relevant functional regions from a set of example proteins. Since a large enough set of examples to perform this kind of analysis is often difficult to obtain, we use a general approach of extending the original training set by exploiting textual annotations. This results in a significantly larger set of labeled examples. We can then train a joint text and sequence classifier over the extended training set, and subsequently project the classifier onto the original sequences to identify the relevant regions. We have shown how we can recover nuclear localization signals using this analysis. The framework takes advantage of recent sequence classification models which are based on analysis of subsequences of the protein and for each position in the sequence, can determine how relevant that position is to predict the class.

We have applied the framework to sub-cellular localization of proteins. We plan to explore alternative ways for combining textual and sequence information using our general approach as well a more thorough analysis of the localization predictions. We also plan to apply our framework to determine relevant regions for other properties of proteins.

1. S. R. Eddy. Multiple alignment using hidden Markov models. In C. Rawlings, editor, *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 114–120. AAAI Press, 1995.
2. A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.
3. K. Karplus, C. Barrett, and R. Hughey. Hidden markov models for

detecting remote protein homologies. *Bioinformatics*, 14(10):846–856, 1998.

4. B. Rost and C. Sander. Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins*, 19(1):55–72, 1994.
5. C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing (PSB)*, Kaua'i, Hawaii, 2002.
6. C. Leslie, E. Eskin, and W. S. Noble. Mismatch string kernels for SVM protein classification. In *Proceedings of Advances in Neural Information Processing Systems 15 (NIPS)*, 2002.
7. C. Leslie, E. Eskin, A. Cohen, and W. S. Noble. Mismatch string kernels for SVM protein classification. Technical report, Columbia University, 2003.
8. R. Nair and B. Rost. Inferring sub-cellular localization through automated lexical analysis. *Bioinformatics*, 18 Suppl 1:S78–S86, Jul 2002.
9. R. Nair and B. Rost. Sequence conserved for subcellular localization. *Protein Sci.*, 11(12):2836–47, Dec. 2002.
10. A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 1998 Conference on Computational Learning Theory*, 1998.
11. K. Tsuda, S. Akaho, and K. Asai. The *em* algorithm for kernel matrix completition with auxiliary data. *Journal of Machine Learning Research*, 4:67–81, 2003.
12. A. Bairoch and R. Apweiler. The SWISS-PROT protein sequence database: its relevance to human molecular medical research. *J. Mol. Med.*, 75:312–316, 1997.
13. W. W. Cohen. Fast effective rule induction. In *International Conference on Machine Learning*, 1995.
14. E. Eskin and S. Snir. A biologically motivated sequence embedding into euclidean space. Technical report, Hebrew University, 2003.
15. E. Eskin, W. S. Noble, Y. Singer, and S. Snir. A unified approach for sequence prediction using sparse sequence models. Technical report, Hebrew University, 2003.
16. V. N. Vapnik. *Statistical Learning Theory*. Springer, 1998.
17. M. Gribskov and N. L. Robinson. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Computers and Chemistry*, 20(1):25–33, 1996.
18. R. Nair, P. Carter, and B. Rost. NLSdb: database of nuclear localization signals. *Nucleic Acids Research*, 31(1):397–9, Jan 2003.