Frequent Subsequence-Based Protein Localization*

Osmar R. Zaïane, Yang Wang, Randy Goebel, and Gregory Taylor

University of Alberta, Edmonton ALB, Canada
{zaiane, wyang, goebel}@cs.ualberta.ca

Abstract. Extracellular plant proteins are involved in numerous processes including nutrient acquisition, communication with other soil organisms, protection from pathogens, and resistance to disease and toxic metals. Insofar as these proteins are strategically positioned to play a role in resistance to environmental stress, biologists are interested in proteomic tools in analyzing extracellular proteins. In this paper, we present three methods using frequent subsequences of amino acids: one based on support vector machines (SVM), one based on boosting and FSP, a new frequent subsequence pattern method. We test our methods on a plant dataset and the experimental results show that our methods perform better than the existing approaches based on amino acid composition.

1 Introduction

Proteins are the molecules that accomplish most of the functions of the living cell. All proteins are composed of linear sequences of smaller molecules called amino acids. There are twenty naturally occurring amino acids. Long proteins may contain a chain of as many as 4500 amino acids. Finding the proteins that make up a creature and understanding their functions is the foundation of explanation in molecular biology [11]. With the introduction of large-scale sequencing, biologists have accumulated an immense volume of raw biological sequences that are publicly available. In order to better understand the functions and structures of these protein sequences, a vitally important problem facing the biology community is to classify these sequences into different families based on the properties of the sequences, such as functions, structures, etc.

Protein sub-cellular localization is a key functional characteristic of proteins. In order to execute a common physiological function, proteins must be localized in the same cellular compartment. Proteins may be localized at various locations within the cell or be transported to the extracellular space. The process through which proteins are routed to their proper sub-cellular localizations is called sub-cellular protein sorting. Protein sorting is the simplest in gram positive prokaryotes, where proteins are only directed to the cytoplasm, the plasma membrane, the cell wall, or secreted to the extracellular space. Gram negative protein localization sites include the cytoplasm, the inner membrane, the

* Research funded in part by the Alberta Ingenuity Funds and NSERC Canada.

J. Li et al. (Eds.): BioDM 2006, LNBI 3916, pp. 35–47, 2006.

periplasm, the outer membrane, and the extracellular space. Sub-cellular localizations in eukaryotic proteins are much more complex due to the presence of membrane-bound organelles. The major location sites for eukaryotic proteins include the plasma membrane, the nucleus, the mitochondria, the peroxisome, the endoplasmic reticulum, the Golgi apparatus, the lysosome, the endosome, and others (such as chloroplasts, vacuoles, and the cell wall in plant cells).

The sub-cellular localization of a protein plays an important role with regard to its function. Knowledge of sub-cellular localization can provide valuable information concerning its possible functions. It can also help in analyzing and annotating sequences of hypothetical or known gene products. In addition, it can influence the design of experimental strategies for functional characterization [5].

Since the number of collected sequences has been rapidly increasing, it is time consuming and costly to approach this problem of predicting the sub-cellular localization of a protein entirely by performing various biological experimental tests. In view of this, it is highly desirable to develop some algorithms to rapidly predict the sub-cellular localizations of proteins.

Herein, we are particularly interested in identifying those proteins that are secreted to the extracellular environment (called *extracellular proteins*), versus proteins localized at various locations within the cell (called *intracellular proteins*) in plants. Extracellular plant proteins are involved in numerous processes including nutrient acquisition, communication with other soil organisms, protection from pathogens, and resistance to disease and toxic metals. Insofar as these proteins are strategically positioned to play a role in resistance to environmental stress, biologists are interested in proteomic tools in analyzing them [26].

A number of methods have been developed in the bioinformatics community for predicting protein sub-cellular localizations. They can be classified into three major approaches based on the features used in the learning algorithms. The first approach is based on "sorting signals", which are short subsequences of approximately 3 to 70 amino acids. For example, SignalP [17, 18] and TargetP [6] use neural networks to identify the sorting signals. The accuracy of SignalP is 68% for human proteins, 70.2% in Eukaryote, 83.7% in E.coli, 79.3% in Gram-negative bacteria and 67.9% in Gram-positive bacteria. TargetP achieves an accuracy of 85% in plants and 90% in non-plant proteins. The second approach is based the amino acid composition. The amino acid composition of a protein sequence refers to the relative frequencies of 20 different amino acids. Each protein is represented by a histogram with 20 bins, regardless of the length of the protein. NNPSL [19] uses neural network and SubLoc [10] uses support vector machines(SVM) to learn the predictors based on amino acid composition. The accuracy of NNPSL is 66% in Eukaryotes excluding plants, and 81% in prokaryotes. The accuracy of SubLoc is 91.4% in prokaryotes and 79.4% on eukaryotes. The third approach, e.g. LOCKey [15] and PA-sub [14], uses the textual information associated with a protein (available in Swiss-Prot [3]) to learn the predictor. Some tools (e.g. PSORT [16]) take an integrative approach by combining several different methods. LOCKey achieves an accuracy of 87% on their test data extracted from Swiss-Prot. PA-sub achieves an overall

accuracy of about 98% in all of their datasets (including animal, archea, fungi, plant, Gram-positive bacteria and Gram-negative bacteria).

Recently, She *et al.* [22] proposed some methods for outer membrane protein classification based on frequent subsequences. Their results have shown that frequent-subsequence-based methods perform better than other methods in the biological domain using precision as a measure. In this paper, we use similar ideas for the problem of extracellular plant protein prediction.

In our work, we use support vector machines as well as boosting using frequent subsequences of amino acids, then combine them with amino acid composition of proteins to improve accuracy. We also introduce a promising new approach FSP specifically designed for frequent subsequences.

2 Predicting Extracellular Proteins

While modeling proteins with histograms representing the amino acid composition has been shown successful [19, 10, 2, 8], we found that the amino acid composition loses discriminant power for plant proteins. Instead we model a protein by a set of frequent subsequences it contains. Our hypothesis is that frequent subsequences of amino acids are better descriptors to discriminate between extracellular and intracellular plant proteins. In this section, after introducing the features used in the training algorithms, we introduce three different methods for extracellular plant protein prediction.

2.1 Feature Extraction

We use *frequent subsequences* as the features for the learning algorithms. A frequent subsequence is a subsequence made up of consecutive amino acids that occurs in more than a certain fraction (*MinSup*) of extracellular proteins. The reason we choose frequent subsequences is based on the following observations:

- Subsequences that appear frequently in extracellular proteins and rarely appear in intracellular proteins have very good discriminative power for identifying extracellular proteins and can be of great interest to biologists.
- It has been known that common subsequences among related proteins may perform similar functions via related biochemical mechanisms [7].
- Frequent subsequences capture the local similarity that may relate to important functional or structural information of extracellular proteins.

There are algorithms for finding frequent subsequences in a set of sequences using generalized suffix trees (GST) [25]. A GST is a trie-like structure designed for compactly representing a set of strings. Each suffix of the string is represented by a leaf in the GST. Each leaf is associated with an index i. The edges are labeled with character strings such that the concatenation of the edge labels on the path from the root to the leaf with index i is a suffix of the ith string in the set. There are algorithms that can construct the GST for a set of strings in linear time [9]. After a GST is constructed, it is traversed in order to find frequent subsequences.

2.2 SVM Method

The first method we use is based on support vector machines (SVM) [24]. SVM is well founded theoretically because it is based on well developed statistical learning theory. It has also been shown to be very effective in real-world applications.

In order to use SVM, the input data have to be in the form of vectors. Each protein sequence is transformed into an n-dimensional vector $\boldsymbol{x} = (a_1, a_2, ..., a_n)$, where n is the number of frequent subsequences found from extracellular proteins, and $a_j (1 \leq j \leq n)$ is the feature corresponding to the ith subsequence. A binary representation is used. If the ith subsequence appears in protein sequence $\boldsymbol{x}$, the value of a_j is set to 1. Otherwise, it is set to 0. For the class label, +1 is used to indicate extracellular proteins and -1 for intracellular proteins.

We can train the SVM using different kernel functions. A kernel function $\Phi(\boldsymbol{x})$ maps the input vector $\boldsymbol{x}$ into a higher dimensional space. Nonlinear separators for the original data can be found by a linear separator in this higher dimensional space. Classical kernel functions include:
Linear Kernel Function: $K(\boldsymbol{x}_i, \boldsymbol{x}) = \boldsymbol{x}_i \cdot \boldsymbol{x}$; Polynomial Kernel Function: $K(\boldsymbol{x}_i, \boldsymbol{x}) = (\boldsymbol{x}_i \cdot \boldsymbol{x} + 1)^d$; and Radial Basic Function(RBF): $K(\boldsymbol{x}_i, \boldsymbol{x}) = exp(-\gamma \parallel \boldsymbol{x}_i - \boldsymbol{x} \parallel^2)$.

2.3 Boosting Method

Boosting is a meta-learning method that has a theoretically justified ability to improve the performance of any *weak classifier*. A weak classifier is an algorithm that, given $\epsilon, \delta > 0$ and access to random examples, can achieve at least slightly better error rate ϵ than random guessing ($\epsilon > 1/2 - \gamma$, where $\gamma > 0$), with a probability $(1-\delta)$. The purpose of boosting is to build a highly accurate classifier by combining many *weak* or *base* hypotheses, each of the weak hypothesis may be only moderately accurate. Various different boosting algorithms have been proposed in the literature [4, 20, 23].

Boosting algorithms work iteratively. During each iteration, a classifier is learned based on a different weighted distribution of the training examples. The main intuition behind boosting algorithms is to increase the weights of the incorrectly classified examples and decrease the weights of the correctly classified examples. This forces the learning algorithm to focus on those examples that are not correctly classified in the next iteration. The algorithm usually stops after a pre-specified number of iterations, or it can stop when some measurement of the quality of the classifier based on certain measurement (such as error rate) starts to deteriorate. The set of classifiers obtained after these iterations are combined together for the final prediction of unseen examples.

In our application of extracellular protein prediction, we use AdaBoost [20] with simple rule-based classifiers as the weak hypotheses. Every rule is a simple check for the presence or absence of a frequent subsequence in a protein primary sequence. Based only on the outcome of this test, the weak hypothesis outputs the prediction and the confidence that each label ("extracellular" or "intracellular") is associated with the protein sequence.

If we denote the possible class label for a protein sequence x by l and define $a \in x$ to represent the fact that subsequence a appears in protein sequence x, the weak hypothesis corresponding to this subsequence has the following form:

$$h(x,l) = \begin{cases} c_{0l} \text{ if a} \in \text{ x} \\ c_{1l} \text{ if a} \notin \text{ x} \end{cases}$$

where the c_{jl} are real numbers. The weak learner searches all possible frequent subsequences. For each subsequence, it calculates the values c_{jl} and assigns a score. Once all the subsequences are searched, the weak hypothesis with the lowest score is returned by the weak learner. In our case, the score is an exact calculation of Z_t (refer to [20] for details). The score is calculated as follows (refer to [21] for details):

Let $X_0 = \{x : w \notin x\}$ and $X_1 = \{x : w \in x\}$. For $j \in \{0,1\}$ and for $b \in \{-1,+1\}$, we calculate the following based on the current distribution D_t:

$$W_b^{jl} = \sum_{i=1}^{m} D_t(i,l)\{x_i \in X_j \wedge Y_i[l] = b\}$$

Z_t is minimized for a particular term by choosing $c_{jl} = \frac{1}{2} ln(\frac{W_{+1}^{jl}}{W_{-1}^{jl}})$ and by setting α_t=1. These settings imply that

$$Z_t = 2 \sum_{j \in \{0,1\}} \sum_{l \in \mathcal{Y}} \sqrt{W_{+1}^{jl} W_{-1}^{jl}}$$

After all frequent subsequences are searched, the weak learner returns the one for which the value of Z_t is the smallest.

2.4 Frequent Subsequence Pattern (FSP) Method

She et al. proposed a rule-based classification based on *frequent patterns*, which have the form $*X_1 * X_2 * ...$, where $X_1, X_2, ...$ are frequent subsequences made up of consecutive amino acids, and "*" is a variable-length-don't-care (VLDC) that can substitute for zero or more letters when matching the pattern against a protein sequence [22]. Their method finds a set of frequent patterns that discriminate outer membrane proteins (OMP) from non-OMPs. In the classification stage, if a protein matches one of the frequent patterns, it is classified as OMP. Otherwise it is classified as non-OMP. We adopt a similar idea in our method, but with the following modification.

Consider a pattern P=$*X_1 * X_2*$ that appears in two different sequences S_1 and S_2 such that X_1 and X_2 are close to each other in S_1 while they are too far apart in S_2. Intuitively, the match in S_1 is more likely to be biologically significant. In our algorithm, we introduce another parameter called $MaxGap$. When matching a pattern against a protein sequence, if the distance (in terms of number of amino acids) of two adjacent subsequences are too far apart, we do not consider it to be a match. For example, if $MaxGap$ is set to be 3, the

pattern "*ABC*DEF*" does not match the sequence "ABCMNOPQDEF", since the gap between subsequence "ABC" and "DEF" is 5 (see Figure 1(a)). However the pattern "*ABC*DEF*" matches the sequence "ABCABCPQDEF", since we can find a way to align them, so that the gap between "ABC" and "DEF" is 2. In this paper, we call the pattern with $MaxGap$ to be *frequent subsequence pattern*, and the pattern without $MaxGap$ to be *frequent pattern*.

Algorithm 1. FSP: Algorithm for Finding Patterns

Input: Training set $D = P \cup N$. (P and N are the sets of extracellular and intracellular proteins)
Output: R a set of patterns in the format of $*X_1 * X_2 * ...$ for predicting extracellular proteins
Parameters: α: rate of weight decrease; δ: threshold total weight; $min\_Znumber$: minimum acceptable Z-number; $MaxGap$: maximum gap between two subsequences.
Method:

```
set the weight of every example in P to 1
pattern set R ← ∅
totalWeight ← TotalWeight(P)
while totalWeight > δ · totalWeight do
  N' ← N, P' ← P
  pattern r ← empty_rule
  while true do
    Choose the subsequence p with the largest Z-number, according to N' and P'
    if Z_number(p) < min_Znumber then
      break
    end if
    append p to r
    for each example t in P' ∪ N' do
      if not Match(t, r, MaxGap) then
        remove t from P' ∪ N'
      end if
    end for
  end while
  R ← R ∪ {r}
  for each example t in P do
    if Match(t, r, MaxGap) then
      t.weight ← α · t.weight
    end if
  end for
  totalWeight ← TotalWeight(P)
end while
return R
```

She et al. use an exhaustive search to build frequent patterns to identify outer membrane proteins by concatenating two or more frequent subsequences [22]. However, since there could be thousands of subsequences found in the training set, an exhaustive search produces an explosive number of candidate patterns. To deal with this problem, we exploit a greedy algorithm to find those patterns. We search for the current best rule and reduce the weights of the positive examples

that are covered by this rule, until the total weight of the positive examples are less than a certain threshold (Algorithm 1). The procedure $Match(t, r, MaxGap)$ in Algorithm 1 is implemented by enumerating all the possible alignment of the subsequences in the pattern r to the sequence t. The pattern r is considered to "match" sequence t, if there is one possible alignment, such that the distances between two adjacent subsequences are all less than $MaxGap$.

Fig. 1. Matching pattern against sequence

The *Z-number* in Algorithm 1 is calculated as follows. Given a rule R and s_R denotes its support, let a_C denote the mean of the target class C, defined as $a_C = |S_C|/|S|$, where S is the current training set and S_C is the subset of S where C is the class label. Let σ_C denote the standard deviation of the target class C. In the binary classification problem, it is calculated as $\sigma_C = \sqrt{a_C(1 - a_C)}$. Using these notions, Z-number is defined as $Z_R = \sqrt{s_R}(a_R - a_C)/\sigma_C$. The Z-number measures how well a rule R discriminates examples of class C [13]. It is similar to the z-test or t-test in statistics. A rule with high positive Z-number predicts the presence of C with high confidence. A rule with high negative Z-number predicts absence of C with high confidence. A rule with Z-number close to zero does not have much power of discriminating examples of class C.

After the set of patterns are generated, we filter them in order to keep those patterns with good predictive power. Only those patterns with support greater than a threshold $MinSup$ and confidence greater than $MinConf$ are kept for predicting unseen protein examples. The prediction process is relatively easy. Given an unseen example t, every pattern r in the pattern set is tested. If there exist a pattern r that matches t, t is predicted to be an extracellular protein, otherwise it is predicted to be an intracellular protein.

3 Experimental Results

Our hypothesis is that frequent subsequences of amino acids are better discriminant than amino acid composition for distinguishing between intracellular and extracellular plant proteins. We compare our methods including SVM based on frequent sequences, boosting based on frequent subsequence, and our frequent subsequence pattern method (FSP) with SVM based on amino acid composition and boosting based on amino acid composition. We also investigate the effect of combining subsequences and amino acid composition in the same classifier.

3.1 Dataset and Evaluation

We tested the performance of our methods on a plant protein dataset that we received from the Proteome Analyst project [14] at the University of Alberta. This dataset contains 3293 proteins, among which 171 are extracellular proteins.

We performed 5-fold cross validation, i.e., each run takes one of the 5 folds as the test set and the remaining 4 folds as the training set. To ensure fair comparisons, all the methods are evaluated using the same folding.

The performance of a classification algorithm is usually evaluated by its *overall accuracy*. However, in our application, overall accuracy is not a good evaluation metric since in our dataset, only about 5% of the proteins are extracellular proteins. A high accuracy (95%) can easily be achieved by classifying every protein to be intracellular. Instead, we choose to use *precision*, *recall* and *F-measure* with respect to the rare class (extracellular proteins) as our evaluation metrics. They are based on the confusion matrix shown in Table 1. Using the notions in Table 1, precision (P) and recall (R) of extracellular class can be defined as:

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN}$$

Table 1. Confusion Matrix

	Actual Extracellular	Actual Intracellular
Predicted as Extracellular	TP	FP
Predicted as Intracellular	FN	TN

The F-measure is a harmonic average of precision and recall: $F = \frac{2PR}{P+R}$.

For all the experiments, the subsequences are obtained by setting the minimum support threshold to be 5%. The numbers of subsequences in each fold are: 2658, 2605, 2532, 2817, 2722 for folds 1 to 5 respectively.

3.2 Experimental Result of SVM

We used the SVM^{light} implementation [12] since it is well-known and has been used extensively in previous research. We tried with three different kernels, including the linear kernel, the polynomial kernel with degree of 2 and the radial basis function kernel with γ=0.005. For each kernel, we tried different values for C (the regularization parameter that controls the trade-off between margin and misclassification error). The best result (in terms of F-measure) using frequent subsequences is **0.804** with a linear kernel. We compared our method with SubLoc [10]. SubLoc uses SVM with amino acid composition as its features. The authors show that SubLoc performs better in terms of accuracy compared with other methods based on amino acid composition. It also performs better than methods based on N-terminal signals. SubLoc is not specifically designed for predicting extracellular proteins, but since its implementation is based on SVM^{light}, we re-implemented it with SVM^{light} and tested it on our dataset.

We tried the same parameter settings as we did in SVM with subsequences. The best result obtained is only **0.522** with a polynomial kernel (d=2) and C=1000.

3.3 Experimental Result of Boosting

In the experiments with boosting, we chose the number of iterations to be 500, 1000 and 2000. The results show that the boosting algorithm is robust with respect to the number of iterations. The best result obtained by boosting using frequent subsequences is **0.729** with 1000 iterations. For the purpose of comparison, we also tried AdaBoost based on amino acid composition. Since the attributes are continuous values in this case, the weak hypothesis used is a single test of whether the composition of an amino acid is above or below some threshold (see [21] for details). The best result obtained is a mediocre **0.574** with 1000 iterations.

3.4 Experimental Result of the FSP Method

For the experiments using the frequent subsequence pattern (FSP) method, there are quite a few parameters to be tuned. In order to tune those parameters, we took a portion of the training examples and tried our algorithm with different parameter settings, then tested the learned model on another portion of the examples. Through trial and error, we identified the following parameter setting: $MinLen$ set to 3, min_gain to 0.1, δ to 0.03 and α to 0.8. The $MinSup$ is set to 5%, $MinConf$ to 80%, and $MaxGap$ to 300. We obtained a precision of **0.765**, a recall of **0.614** and an F-measure of **0.681**.

Even though SVM based on frequent subsequences achieves the best experimental result, there are some advantages in using the FSP method. The reason is that the decision functions learned by SVM algorithms are difficult for people to understand. The discovered hyperplane is difficult to manipulate. However, the decision rules found by the FSP method can be easily interpreted and modified by human experts. Figure 2 shows some examples of the rules found by the FSP method. Biologists can easily read these rules and determine whether they are biologically meaningful. They can also incorporate their biological knowledge and modify the patterns, e.g., by adding or removing subsequences in the patterns, to get even better classification models. This study is currently in progress.

```
IF (sequence contains *CKN*CGPGHGIS*) THEN (extracellular)
IF (sequence contains *YWGQNG*EIN*) THEN (extracellular)
IF (sequence contains *QVY*AGH*NVT*) THEN (extracellular)
...
ELSE (intracellular)
```

Fig. 2. Examples of patterns found by the FSP method

3.5 Combining Frequent Subsequences and Amino Acid Composition

It is clear that the methods based on frequent subsequences perform better than those based on amino acid composition. However, we can still take advantage of the information represented in the amino acid composition histograms by combining these two features. We investigated this possibility. Interestingly, we found that SVM does not improve at all. The result shows that there is no obvious benefits of combined features for SVM. In other words, SVM could not take advantage of the additional information. In the case of the RBF kernel, SVM based on combined features deteriorated. The additional data (amino acid composition) created noise.

Contrary to SVM, the performance of boosting (measured by F-measure) can be improved significantly by combining frequent subsequences and amino acid composition. As can be seen in Table 2 and Table 3 Boosting using combined features gives a better result than the best result of SVM with frequent subsequences reaching **0.831**. Boosting better exploits this additional data regarding the amino acid composition when added to the frequent subsequences.

Table 2. AdaBoost classification with combined features

Number of iterations	Recall	Precision	F-measure
500	0.685	0.967	0.802
1000	0.717	0.989	**0.831**
2000	0.708	0.989	0.826

Table 3. Comparison of AdaBoost based on different features

Number of iterations	Combined feature	Subsequence	Composition
500	0.802	0.714	0.562
1000	**0.831**	**0.729**	**0.574**
2000	0.826	0.726	0.548

Since amino acid composition is represented by fractional numbers (i.e. the histogram), there is no easy way to merge frequent subsequences and amino acid composition in the FSP method. Thus, we did not combine amino acid compositions in our FSP algorithm. However, a new model to represent this information is worth investigating.

For cross comparison, we chose the best (in terms of F-measure) result generated by each algorithm (i.e., 0.804 for SVM with subsequences, 0.729 for boosting with subsequences, 0.522 for SVM with amino acid composition, 0.574 for boosting with amino acid composition). The comparison of different algorithms is shown in Figure 3. Our methods based on frequent subsequences are better than methods based on amino acid composition. In particular, Boosting with a combination of frequent subsequences and amino acid composition performs the best among the different approaches reaching an F-measure of **0.831**.

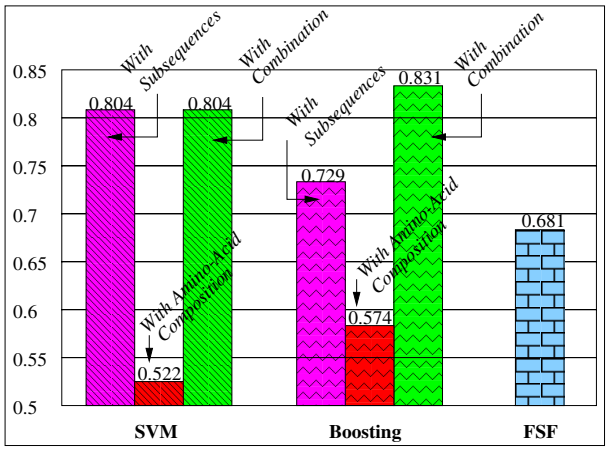

Fig. 3. F-measures of different algorithms

4 Conclusion and Future Work

We present in this paper several methods for identifying extracellular plant proteins using frequent amino acid subsequences. Our experimental results show that our methods perform better than amino acid composition-based methods. The best result is actually achieved by combining frequent amino acid subsequences and amino acid composition using Boosting. Combining these two features is not always beneficial. It was indeed detrimental in the case of SVM.

Even though the experimental results show SVM and boosting based on frequent subsequences as being the best approaches, there are advantages in using our new FSP method. The main reason being that contrary to SVM for example, the decision functions of the FSP method are easily readable rules that can be easily understood, interpreted and edited by human experts. Moreover, FSP is extendable. While it can not accommodate amino acid composition for the moment, additional information such as location of frequent subsequences, and constraints on their sizes could be combined in the algorithm.

There are a number of directions for possible future research. First of all, we only use the protein primary sequences for training the predictor of extracellular proteins. If additional properties of proteins (e.g., secondary structures, functions) are available, future research can take these characteristics into account to make a more accurate prediction.

With respect to frequent subsequences of amino acid, one important feature is the location of the subsequence within the protein. Biologists believe that the position of a frequent subsequence, in the beginning, the end, or other, within the protein can provide some indication regarding the protein itself. We are currently investigating the combination of frequent subsequences, amino acid composition, and the relative subsequence positions to build a more robust classifier. In particular, we are dividing a protein sequence into percentiles

(50%, 25%, and 10%) and labeling the relative position of a frequent subsequence by the portion in the protein where the subsequence starts. One good model that lends itself to this type of combinations is the associative classifier [1]. Based on associations rules, classification rules can be learned from proteins modeled into transactions of features. These rules are also easily understood and potentially modifiable by human experts in include additional domain knowledge.

References

1. M.-L. Antonie, O. R. Zaïane, and A. Coman. *Mining Multimedia and Complex Data*, chapter Associative Classifiers for Medical Images, pages 68–83. Lecture Notes in Artificial Intelligence 2797. Springer-Verlag, 2003.
2. M. Bhasin and G. Raghava. Eslpred: SVM-based method for subcellular localization of eukaryotic proteins using dipeptide composition and psi-blast. *Nucleic Acids Research*, 32:W414 – W419, July 2004.
3. B. Boeckmann, A. Bairoch, R. Apweiler, M.-C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. The Swiss-Prot protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Research*, 31:365–370, 2003.
4. W. Cohen and Y. Singer. A simple, fast and effective rule learner. In *Proceedings of Annual Conference of American Association for Artificial Intelligence*, pages 335–342, 1999.
5. F. Eisenhaber and P. Bork. Wanted: subcellular localization of proteins based on sequence. *Trends in Cell Biology*, 8:169–170, 1998.
6. O. Emanuelsson, H. Nielsen, S. Brunak, and G. von Heijne. Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *Journal of Molecular Biology*, 300:1005–1016, 2000.
7. K. A. Frenkel. The human genome project and informatics. *Communications of the ACM*, 34(11):41–51, 1991.
8. A. Garg, M. Bhasin, and G. Raghava. Support vector machine-based method for subcellular localization of human proteins using amino acid compositions, their order, and similarity search. *Journal of Biological Chemistry*, 280(15):14427 – 14432, April 2005. Epub 2005 Jan 12.
9. D. Gusfield. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, 1997.
10. S. Hua and Z. Sun. Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 17(8):721–728, 2001.
11. L. Hunter. *Artificial Intelligence and Molecular Biology*. AAAI Press, 1993.
12. T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer, 2002.
13. M. V. Joshi, R. C. Agarwal, and V. Kumar. Mining needles in a haystack: Classifying rare classes via two-phase rule induction. In *Proceedings of ACM SIGMOD Conference*, pages 91–102, Santa Barbara, CA, 2001.
14. Z. Lu. Predicting protein sub-cellular localization from homologs using machine learning algorithms. Master thesis, 2002. Department of Computing Science, University of Alberta.
15. R. Nair and B. Rost. Inferring sub-cellular localization through automatic lexical analysis. In *Proceedings of the tenth International Conference on Intelligent Syetems for Molecular Biology*, pages 78–86. Oxford University Press, 2002.

16. K. Nakai. A knowledge base for predicting protein localization sites in eukaryotic cells. *Genomics*, 14:897–911, 1992.
17. H. Nielsen, J. Engelbrecht, and S. Brunak. A neural network method for identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *International Journal of Neural Systems*, 8:581–599, 1997.
18. H. Nielsen, J. Engelbrecht, S. Brunak, and G. von Heijne. Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Engineering*, 10(1):1–6, 1997.
19. A. Reinhardt and T.Hubbard. Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Research*, 26(9):2230–2236, 1998.
20. R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
21. R. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2):135–168, 2000.
22. R. She, F. Chen, K. Wang, M. Ester, J. L. Gardy, and F. S. L. Brinkman. Frequent-subsequence-based prediction of outer membrane proteins. In *Proceedings of ACM SIGKDD Conference*, Washington, DC, USA, 2003.
23. K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *Proceedings of Intl. Conference on Machine Learning*, pages 983–990, 2000.
24. V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.
25. J. Wang, G. Chirn, T. Marr, B. Shapiro, D. Shasha, and K. Zhang. Combinatorial pattern discovery for scientific data: Some preliminary results. In *Proceedings of ACM SIGMOD Conference*, Minnesota, USA, 1994.
26. Y. Wang. EPPdb: a database for proteomic analysis of extracytosolic plant proteins. Master thesis, 2004. Department of Computing Science, University of Alberta.