Contrast Pattern Mining with Gap Constraints for Peptide Folding Prediction*

Chinar C. Shah, Xingquan Zhu, Taghi M. Khoshgoftaar, Justin Beyer

Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431
{cshah2, xzhu3, khoshgof, jbeyer}@fau.edu

Abstract

In this paper, we propose a peptide folding prediction method which discovers contrast patterns to differentiate and predict peptide folding classes. A contrast pattern is defined as a set of sequentially associated amino acids which frequently appear in one type of folding but significantly infrequent in other folding classes. Our hypothesis is that each type of peptide folding has its unique interaction patterns among peptide residues (amino acids). The role of contrast patterns is to act as signatures or features for prediction of a peptide's folding type. For this purpose, we propose a two phase peptide folding prediction framework, where the first stage is to discover contrast patterns from different types of contrast datasets, followed by a learning process which uses all discovered patterns as features to build a supervised classifier for folding prediction. Experimental results on two benchmark protein datasets will indicate that the proposed framework can outperform simple secondary structure prediction based approaches for peptide folding prediction.

1. Introduction

Protein/peptide (peptides are small proteins which are short chains of amino acids with an average length of 20 residues) folding is the physical process by which the local and global interactions among amino acids enforce a sequence to fold into unique three-dimensional structures, through which the proteins/peptides can carry out their diverse cellular functions (Lesk 2002), such as breaking down starch chains into smaller sugar molecules (Amylases) or carrying oxygen (hemoglobin). The function of the protein or peptide crucially relies on their 3-D conformation structure, and changes made to such structure will most likely lead to changed or failed functions even if the primary structure (amino acid sequence) remains the same. Existing research (Qian & Sejnowski 1998, Kabsch & Sander 1983, Ptitsyn & Finkelstein 1982) suggested that 3-D structures of a protein or peptide are solely a function of its amino acid sequence, although exact formula of the function remains unknown. It has been commonly agreed that due to local amino acid interactions mediated by hydrogen bonds, amino acid chains are organized into a certain types of local regular structures, known as alpha-helices (α-Helix), beta-pleated sheets (β-Sheet), and random coils (the remaining part), commonly referred to as secondary structures (Kabsch & Sander 1983, Ptitsyn & Finkelstein 1982). The one dimensional amino acid sequence spontaneously folds into 3-D structures, which eventually determine biological functions of the molecule. In practice, although finding such 3-D structure is possible with complex techniques such as X-ray crystallography, the gap between the rapid growth of the number of protein sequences and the unavailability of their structure information requires effective and economical solutions to predict structures automatically. Formally, the problem of protein/peptide folding prediction is to predict 3-D folding type of an amino acid sequence, without complex and time-consuming processes like X-ray crystallography or protein nuclear magnetic resonance spectroscopy (NMR), such that we can understand protein/peptide structures, conformations, and functions directly from their amino acid sequences.

If amino acid sequences contain sufficient information to determine three-dimensional structures, it should be possible to devise an algorithm to predict structures from the amino acid sequence. Unfortunately, this has proved elusive (Lesk 2002). Instead, a large body of work has focused on less ambitious goals such as secondary structure prediction (Frishman & Argos 1997) and folding type recognition (Ding & Dubchak 2001). For example, to understand the relationship between an amino acid sequence and its corresponding structures, general distance functions or support vector machines are used to classify protein folding into five categories: All-Helix , All-*Sheet*, *Helix/Sheet*, *Helix+Sheet, and small proteins.* Ding (Ding & Dubchak 2001) further extended folding types to 27 classes, with the best classification accuracy close to 50%. Similarly, the problem of peptide folding prediction is to classy folding type of a short amino acid chain into one of the following seven classes: All-helix(H), All-Sheet(E), All-Coil(C), Helix-Sheet(HE), Helix-Coil(HC), Sheet-Coil(BC), and Helix-Sheet-Coil(HEC). More specifically, a peptide's folding is identified as All-Helix (H) if and only if all its residues' secondary structures are helix (the same logic applies to E and C as well), and a complex folding like "HE" means that the secondary structures of a peptide's residues are mixture of Helix and Beta sheet.

*Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved. This research has been supported by the National Science Foundation of China (NSFC) under Grant No.60674109 and the US National Science Foundation (NSF) under Grant No. OISE-0730065.

For automatic protein secondary structure prediction, a sliding window can be used to move along the amino acid sequence to generate examples for model training. For sequences with known secondary structures (e.g., from DSSP label (Kabsch & Sander 1983)), we can move a small window along the sequence, with residues covered by the window taking as attribute values and the secondary structure corresponding to the central residue of the window as the class label of the instance. The generated training instances can be used for supervised learning algorithms to build models for secondary structure recognition.

Because peptides are small proteins, the prediction of the peptide folding can also be achieved through secondary structure prediction. For example, given a peptide sequence "KPECPV", one can predict secondary structure of each residue, and if the results are "HHH---" which is a mixture of Helix and Coil, the folding type of the peptide will be identified as "HC". Unfortunately, such a secondary structure based approach has two major drawbacks: First, most secondary structure prediction methods are window-based which adopt a sliding window to predict secondary structures of the central residue. For short amino acid chains like peptides (the average length of peptides is about 20), the sliding windows will not be able to predict the secondary structure of the beginning and the ending part of the sequence effectively. Secondly, because secondary structure predictions are highly inaccurate (especially for β-Sheet and when homology sequences are not available), errors made by secondary structure prediction will propagate to folding prediction and severely deteriorate system performances.

Motivated by the above observations, we propose in this paper a contrast pattern mining based peptide folding prediction method. Our hypothesis is that each folding type has its own unique patterns which associate different folding types to local amino acid sequences. This hypothesis is rooted from the well accepted presumptions that each residue's secondary structure is appreciably correlated with the local amino acid sequences and that these correlations can be used to predict secondary structures, or contribute to other tertiary structure prediction (Bowie et al 1991). Instead of solely relying on the secondary structure prediction results to identify peptide folding, our approach directly discovers contrast patterns for three types of folding: all-Helix, all-Sheet, and all-Coil, where a contrast pattern is defined as a set of sequentially associated amino acids. To ensure flexibilities of pattern matching, we allow that each pattern bears a certain degree of freedom in its appearances (commonly referred to as gap constraints (Ji et al. 2005, Zhu and Wu 2007)). For example, $K\phi_0^2 P\phi_0^2 E$ defines a pattern with three letters (amino acids) K, P, and E, and the gap constraint between any two letters are [0, 2], which means that the appearances of the consecutive pattern letters should be within the range 0 and 2. Sequences like $\underline{K}\underline{P}V\underline{E}V$ and $\underline{K}SL\underline{P}K\underline{E}V$ would both match the pattern (marked with underscores).

The remaining part of the paper is structured as follows. Section 2 addresses contrast pattern mining from protein sequences, followed by Section 3 which uses contrast patterns to build supervised classifiers for peptide folding prediction. Section 4 reports experimental results on two protein datasets (RS126 (Rost & Sander 1993) and BC513 (Cuff and Barton 2000)) which contain 126 and 513 protein sequences respectively, and we conclude in Section 5.

2. Contrast Pattern Mining

A contrast pattern defines a set of sequentially associated pattern letters which uniquely match a group of examples. The similar problem has been addressed in other research (Ji et al. 2005) which identifies minimal distinguishing subsequences from two datasets with gap constraints. In this Section, we propose to identify patterns from sequence databases with different secondary structures. The system flowchart is shown in Figure 1.

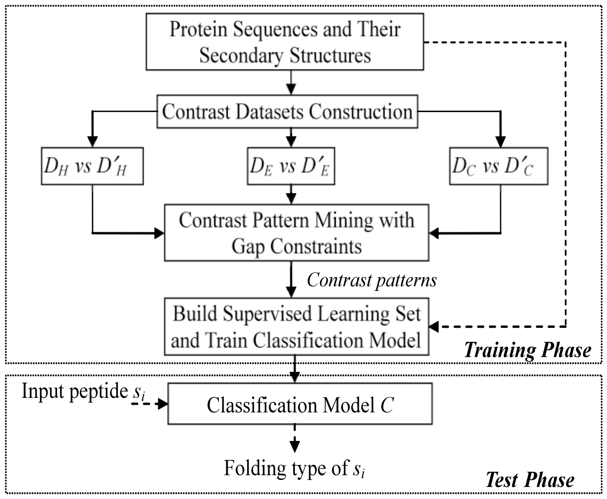
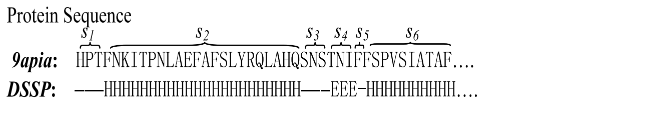

Protein Sequences and Their Secondary Structures
Contrast Datasets Construction
D_H vs D'_H
D_E vs D'_E
D_C vs D'_C
Contrast Pattern Mining with Gap Constraints
Contrast patterns
Build Supervised Learning Set and Train Classification Model
Training Phase
Input peptide s_i
Classification Model C
Folding type of s_i
Test Phase

Figure 1: System Framework


Protein Sequence
s_1
s_2
s_3
s_4
s_5
s_6
9apia: HPTFNKITPNLAEFAFSLYRQLAHQSNSTNIFFSPVSIATAF....
DSSP: ——HHHHHHHHHHHHHHHHHHHHHHHH——EEE-HHHHHHHHHH....
Secondary Structures

$D_H = \{s_2, s_6, \ldots\} = \{$FNKITPNLAEFAFSLYRQLAHQ , FSPVSIATAF, …$\}$
$D'_H = \{s_1, s_3, s_4, s_5 \ldots\} = \{$HPT, SNS, TNI, F,…$\}$
$D_E = \{s_4, \ldots\} = \{$TNI, …$\}$
$D'_E = \{s_1, s_2, s_3, s_5, s_6 \ldots\} = \{$HPT, FNKITPNLAEFAFSLYRQLAHQ,TNI,FSPVSIATAF, .$\}$
$D_C = \{s_1, s_3, s_5, \ldots\} = \{$HPT, SNS, F,…$\}$
$D'_C = \{s_2, s_4, s_6, \ldots\} = \{$FNKITPNLAEFAFSLYRQLAHQ,TNI, FSPVSIATAF,…$\}$

Figure 2: An example of contrast dataset construction

The whole system consists of two phases: training phase and test phase. The training phase first mines contrast patterns from a set of amino acid sequences where each contrast pattern taking the forms as a set of sequentially associated amino acids bounded by a gap constraint. The second step of the training phase uses discovered contrast patterns to build a supervised learning set, from which a classification model can be constructed to predict the folding type of each short amino acid chain (peptide) s_i.

2.1 Contrast Datasets

The first step of the training phase is to transfer training protein sequences into three pairs of contrast datasets (D_H *vs.* D'_H, D_E *vs.* D'_E, and D_C *vs.* D'_C), from which the mining algorithm can discover contrast patterns. Because each protein sequence normally contains several hundred residues but a peptide only has about 20 residues on average, our process of building contrast datasets is to parse each protein sequence into a set of consecutive subsequences, according to the secondary structures of the subsequence. An example of the process is demonstrated in Figure 2, where the first and the second row report the sequence information of protein 9apia and its second structures defined by DSSP (Kabsch & Sander 1983). In order to build contrast datasets to mine patterns, each protein sequence is parsed into a set of non-overlapping Maximum Consecutive Subsequences (MCS), with each MCS defined as follows:

A maximum consequence subsequence is a consecutive amino acid sequence segment, where all residues of the segment have the same type of secondary structure which is different from the secondary structures of the residues next to the segment (on both sides).

Based on the above definition, given protein sequence (9apia) in Figure 2, we can partition it into 6 MCSs (s_1, s_2, .., s_6) each of which containing a set of amino acids with the same type of secondary structures (α-*Helix*, β-*Sheet*, and *Coil*). Each segment s_i is further allocated to dataset D_X or D'_X, depending on whether the secondary structure of s_i belongs to X or not. After that, each pair of datasets D_X and D'_X contains a set of short amino acid chains with counter properties (the MCSs with/without one type of folding). D_x and D'_X thus constitute a pair of contrast datasets, from which a set of contrast patterns can be discovered.

The approach in Figure 2 parses each long protein sequence into short consecutive subsequences each of which has the same type of secondary structure and is taken as a peptide. Consequently, we can regard each dataset D_X consisting of a set of peptides with similar folding types. The reason we are using this approach is because it is difficult to find a large number of peptides with known secondary structures, but finding a number of proteins with known secondary structures is relatively easy. Meanwhile, because a peptide consists of a short chain of amino acids, the approach proposed here can be applied to any peptide datasets directly.

2.2 Contrast Pattern Mining with Gap Constraints

After the construction of contrast datasets, the next step is to mine patterns from each pair of contrast datasets. For this purpose, we propose an Apriori-based (Agrawal et al 1993) Contrast Pattern Mining (CPM) algorithm with gap constraints (Figure 3), which gradually generates and grows candidate patterns, followed by a validation process to check whether a candidate satisfies users' specifications (both gap and support value requirements).

```
ContrastPatternMiningWGapConstraint ($\phi_a^b$, $D_A$, $D_B$, Min_sup, Max_sup)

Parameters: (1) $\phi_a^b$: Gap constraints; (2) $D_A$ and $D_B$: A pair of contrast
datasets; and (3) Min_sup and Max_sup: minimal and maximal frequency for
patterns in $D_A$ and $D_B$ respectively.
Output: Contrast Pattern Set ($CP$)
 1.  $CP \leftarrow \phi$; $C_2 \leftarrow \phi$; $L_2 \leftarrow \phi$; $C_n \leftarrow \phi$; $L_k \leftarrow \phi$, $k=1,2,3\ldots$
 2.  $L_1 \leftarrow$ Check sequences in $D_A$ and find single amino acid with its
     frequency larger than $Min_{sup}$
 3.  For any two amino acids $l_i$ and $l_j$ in $L_1$, generate one candidate pattern
 4.          $C_2 \leftarrow C_2 \cup l_i \phi_a^b l_j$;        $l_i \in L_1$ and $l_j \in L_1$
 5.  EndFor
 6.  For each pattern $p_i$ in $I_2$
 7.      $L_2 \leftarrow L_2 \cup p_i$, if $p_i$'s frequency in $D_A$ is larger than $Min_{sup}$
 8.          $CP \leftarrow CP \cup p_i$, if $p_i$'s frequency in $D_B$ is less than $Max_{sup}$
 9.  EndFor
10.  $k \leftarrow 2$
11.  While ($L_k$ is nonempty)
12.      For each pattern $p_i$ in $L_k$
13.          For each amino acid $l_j$ in $L_1$
14.              Generate candidate $c_m \leftarrow p_i \phi_a^b l_j$
15.              If all length-$k$ subsequences of $c_m$ belong to $p_i$
16.                  If $c_m$'s frequency in $D_A$ is larger than $Min_{sup}$
17.                      $L_{k+1}=L_{k+1} \cup c_m$
18.                      If $c_m$'s frequency in $D_B$ is less than $Maxs_{up}$
19.                          $CP \leftarrow CP \cup c_m$
20.                      EndIf
21.                  EndIf
22.              EndIf
23.          EndFor
24.      EndFor
25.  $k \leftarrow k+1$
26.  EndWhile
27.  Output contrast pattern set $CP$
```

Figure 3: Contrast pattern mining with gap constraints (CPM)

The CPM mining process in Figure 3 takes the following three inputs and parameters: (1) ϕ_a^b: the gap requirements for pattern letters; (2) a pair of contrast datasets D_A and D_B (D_H *vs.* D'_H, D_E *vs.* D'_E, or D_C *vs.* D'_C); and (3) the minimal (Min_{sup}) and maximal (Max_{sup}) support values which specify a contrast pattern's frequency in D_A and D_B respectively. All contrast patterns met users' requirements are output through the pattern set CP.

As the first step of the mining process, CPM checks dataset D_A and any single amino acid with its frequency higher than Min_{sup} is put into set L_1 (length-1 set). Then CPM joins any two amino acids l_i and l_j in L_1 to generate a set of length-2 candidates (C_2), followed by a validation process to check each candidate's frequency in D_A and D_B respectively (Steps 6 to 9). After that, CPM starts a looping process to grow the length of the patterns one amino acid a time until no candidate in the current round k can have its frequency larger than Min_{sup} in D_A, *i.e.*, no need to generate more candidates (Steps 11 to 26).

For each round k of the looping process, L_k contains all frequent patterns *w.r.t.* dataset D_A. CPM first generates a set of candidates by appending each amino acid in L_1 to each pattern in L_k to produce a length-k+1 candidate c_m. According to the Apriori principle (Agrawal et al 1993), if a length-$k+1$ pattern p_i is frequent in dataset D, any of p's length-k subpattern must be frequent in D as well. Accordingly, for each length-k+1 candidate c_m, CPM checks its length-k subsequences, and if any of c_m's length-k subsequence does not belong to L_k, it means that c_m cannot possibly become a frequent pattern (Step 15). For each candidate c_m with its length-k subsequences belonging to L_k, CPM checks its frequency in D_A and assigns it to frequent pattern set L_{k+1}, as long as c_m's frequency in D_A is larger than Min_{sup}. In addition, CPM continuously checks each frequent pattern c_m's frequency in D_B and takes c_m as a contrast pattern if its frequency in D_B is less than Max_{sup} (Steps 18 to 19).

3. Peptide Folding Prediction

For each pair of contrast datasets D_X and D'_X, we will sequentially carry out mining process in Section 2.2 by setting $\{D_A \leftarrow D_X, D_B \leftarrow D'_X\}$ and $\{D_A \leftarrow D'_X, D_B \leftarrow D_X\}$ respectively, which will mine a set of patterns frequent in D_X *w.r.t.* Min_{sup} but infrequent in D'_X *w.r.t.* Max_{sup}, and patterns frequent in D'_X *w.r.t.* Min_{sup} but infrequent in D_X *w.r.t.* Max_{sup}. In addition, since we have three pairs of contrast datasets (D_H *vs.* D'_H, D_E *vs.* D'_E, and D_C *vs.* D'_C), we will repeat the mining process 6 times like: (1) $\{D_A \leftarrow D_H, D_B \leftarrow D'_H\}$; (2) $\{D_A \leftarrow D'_H, D_B \leftarrow D_H\}$; … (5) $\{D_A \leftarrow D_C, D_B \leftarrow D'_C\}$; and (6) $\{D_A \leftarrow D'_C, D_B \leftarrow D_C\}$. After that, all contrast patterns are collected to construct a set of training examples, from which a supervised classifier can be built for peptide folding prediction.

The procedure of training example construction takes a set of protein sequences S with known secondary structures and a contrast pattern set CP with n patterns as input to generate *trainInstanceNum* labeled training instances, each having $n+20$ dimensional features and one class label. In order to generate a single training instance I_x, our algorithm randomly chooses a protein sequence s from S, and then randomly selects a small consecutive subsequence s_i from s (Steps 3 to 4). We calculate the number of times each amino acid appears in s_i, and use these values as the first 20 dimensional features (because peptide sequences consist of 20 types of amino acids). After that, we sequentially check whether a contrast pattern c_j (j=1,…,n) in CP match s_i or not. If c_j appears in s_i, we set feature f_{20+j} as 1, otherwise, we set f_{20+j} as 0. Therefore, each subsequence s_i will have 20+n dimensional features in total. The class label of the instance I_x is determined by the secondary structures of all residues in s_i. If all residues belong to α-Helix (All-Helix), we label I_x as "H". The same logic can be applied to other classes, *e.g.*, "E", "C", "HE", "EC", "HC", and "HEC" as well. An example of the training example construction is demonstrated in Figure 5. Each iteration of the above process generates one training example, and the algorithm repeats until *trainInstanceNum* examples are generated.

Notice that random consecutive subsequence selection process on Step 4 has a very small chance of selecting a subsequence with all its residues belonging to one type of secondary structure. Consequently, we adjust the instance construction process by first selecting all maximum consecutive subsequences from protein sequences. The purposes is to balance training examples for folding classes like "H", "E", and "C", such that a learner can receive good performances over all classes of examples. In addition, when selecting a subsequence s_i on Step 4, the length of s_i is randomly chosen between 5 and 25 (*i.e.*, typical peptide length in practice).

After the construction of the learning set, the problem of peptide folding is well defined as a classification task, learning algorithms such as Naïve Bayes (Domingos P., and Pazzani 1997), Neural Networks, Support Vector Machines (Cristianini & Shawe-Taylor 2000), or C4.5 decision trees (Quinlan 1993) can be applied to build a classification model to predict folding types of a peptide. In our experiments, we use Naïve Bayes classifiers frequently mainly because of its time efficiency. For example, it can easily take more than 10 hours for SVM to build one classifier but NB only takes about 10 seconds on the same data.

4. Experimental Results

4.1 Experimental Setting

We validate our algorithms on two commonly used protein datasets (RS126 and CB513). The first one is the dataset of 126 protein chains (RS126) (Rost & Sander 1993). This is a non-homologous dataset according to the definition of Rost & Sander where no two proteins in the set share more than 25% sequence identify over a length of more than 80 residues. The second dataset has 513 protein chains (Cuff and Barton 2000), and almost all the sequences in the RS126 set are included in the CB513 set. For both datasets, we performance 10 times 10-fold cross validation, and only sequences in the training set are used to build contrast datasets and mine contrast patterns.

The whole system is implemented in Java with an integration of WEKA data mining tool (Witten & Frank 2005) which provides extensive data mining functions for system development. The majority results are based on Naïve Bayes classifiers, unless specified otherwise. Although Support Vector Machines are reported to outperform other learners (Ding & Dubchak 2001), we found SVM perform extremely slow especially for a large number of training examples with high dimensional features. Since we are mainly interested in the relative improvement of the proposed algorithm CPM over simple solutions, we believe that conclusions drawn here are valid for SVM and other learners as well.

For comparison purposes, we implement a simple secondary structure prediction algorithm (SmpSnd) with a window size 13 (Qian & Sejnowski 1993). The peptide folding prediction results based on the secondary structure prediction is denoted by "SmpFld" in all tables and figures.

4.3 Peptide Folding Prediction Accuracy Assessment

For comparison purposes, we implement a simple secondary structure prediction based peptide folding prediction method (SmpFld). We first build a simple secondary structure predictor (SlmSnd) based on the framework in Figure 1. For each peptide *p*, we use SlmSnd to predict secondary structures of its residues. After that, we check whether the predicted secondary structures form a correct folding type for *p*. If the predicted secondary structures change the folding type of *p*, we say SmpFld incorrectly predicts the folding type of *p*. In Table 2, we report the peptide folding prediction results for three folding classes (All-Helix, All-Sheet, and All-Coil), where training and test peptides are all maximum consecutive subsequences (e.g. no random selection process to generate
complex folding types like "HE" or "EC").

Table 1 reports the peptide folding prediction results, where four measures, True Positive Rate (*TPR*), False Positive Rate (*FPR*), F-measure, and overall prediction accuracy are used to assess the algorithm performances. The formal definitions of each measure are given by Eqs. (1) to (5). The F-measure is the harmonic mean of precision and recall which can be used as a single measure of performance of the test (the higher the F-measure value, the better the algorithm performs). For all results in Table 1, the gap constraints are set to [0, 2] and Min_{sup} and Max_{sup} are specified such that the total number of mined contrast patterns is always less than 500.

$$TPR = \frac{\#\ of\ residues\ in\ X\ \&\ correctly\ predicted\ as\ X}{\#\ of\ residues\ in\ X} \quad (1)$$

$$FPR = \frac{\#of\ residues\ in\ \sim X\ but\ predicted\ as\ X}{\#of\ residues\ in\ \sim X} \quad (2)$$

$$precision = \frac{\#\ of\ residues\ correctly\ predicted\ as\ X}{\#\ of\ residues\ predicted\ as\ X} \quad (3)$$

$$accuracy = \frac{\#\ of\ correctly\ predicted\ residues}{\#\ of\ predicted\ residues} \quad (4)$$

$$F-measure = 2 \cdot \frac{precision \cdot TPR}{(precision + TPR)} \quad (5)$$

The results in Table 1 indicates that when considering three folding classes (All-Helix, All-Sheet, and All-Coil) only, the accuracy of CPM is significantly better than simple secondary structure based approach, where the average prediction accuracies of CPM over all three classes are about 70%, which are more than double the accuracy of SmpFld. Actually, the accuracy of SmpFld is even worse than random guessing.

In Figure 4, we further report the prediction accuracies over all seven folding classes (H, E, C, HE, HC, EC, and HEC). As we have addressed in Section 3, the training examples for HE, HC, EC, and HEC are generated from random consecutive subsequences, and the class labels are based on the secondary structures of all residues of the subsequence (based on DSSP label). For comparison purposes, we also report algorithm performance by using different types of learners (Naïve Bayes, C4.5 decision trees, RBF neural networks, and SVM). Because high dimensional features significantly increase the time complexity for C4.5, RBF, and SVM, the number of contrast patterns selected in Figure 6 is limited to less than three hundreds.

The results in Figure 4 indicate that when considering all seven peptide folding classes, the average prediction accuracy of CPM decreases from 70% to about 51%, and on the other hand, the accuracy of SmpFld increases from 30% to around 40% (CB513), which is still significantly worse than CPM. The accuracy increase of SmpFld is due to the fact that secondary structure based peptide folding prediction is less sensitive to complex folding types like HEC. Because HEC type folding consists of a mixture of three types of residues, it is possible that one or multiple incorrect predictions of the secondary structure may still preserve the folding types of the peptide. For example, if the genuine secondary structure of a peptide is "---HHH--EE", incorrect secondary structure predictions like "EE-H----HH" or "---HHHEEEE" would still make a correct folding prediction. Consequently, adding complex peptide folding classes would increase the prediction accuracy of SmpFld. On the other hand, Since CPM uses contrast patterns customized for three types of folding (H, E, and C) to build peptide folding classifiers, patterns discovered by CPM may not effectively capture complex folding types like HEC. As a result, a decrease of the overall prediction accuracy may be observed.

The accuracy comparisons from different types of learners including C4.5, RBF, and SVM further assert the effectiveness of contrast patterns in supporting supervised learning for peptide folding prediction. Similar to the conclusions drawn from the previous research work (Ding & Dubchak 2001), given same training and test data, SVM receives relatively better performances than NB, C4.5, and RBF. Meanwhile, we observed that for our particular problem, Naïve Bayes classifiers perform much better than C4.5 decision trees and NB's results are comparable to RBF neural networks most of the time. This is interesting, as we know that NB classifiers consider attributes conditionally independent given the class label, and some contrast patterns are highly correlated because they may be grown from the same superset. For example, patterns $A\phi_a^b K\phi_a^b G$ and $A\phi_a^b K\phi_a^b M$ are both grown from one superset $A\phi_a^b K$, so they are highly correlated instead of being independent. However, just like observations made by Domingos (Domingos & Pazzani 1997), NB performs reasonably well on many real-world datasets where attribute independence is known not to hold, possibly because the assumption of conditional independence is still valid for the majority of the attributes. Considering high efficiency and relatively good performances of NB in solving our problem, we strongly suggest interested readers to consider this learner in their future research work.

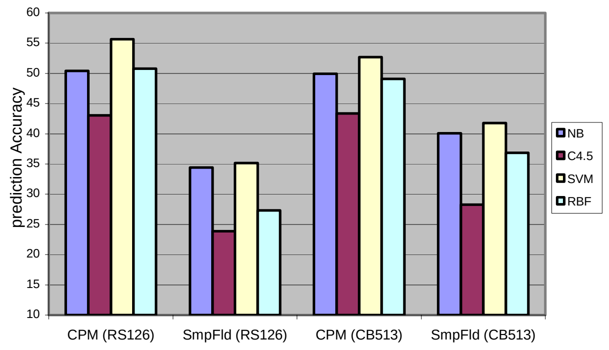


Figure 4: Peptide folding prediction accuracies (7 classes)

5. Conclusions

In this paper, we presented a contrast pattern based peptide folding prediction algorithm, where a contrast pattern is defined as a set of sequentially correlated amino acids which frequently appear in one type of folding but significantly infrequent in other folding classes. Our hypothesis is that different types of peptide folding contain unique interactions among amino acids of the sequences, and finding such patterns can be beneficial for predicting folding types of a peptide from its primary structure. In order to discover contrast patterns, we first constructed three pairs of contrast datasets each containing subsequences with or without one type of secondary structure. We mine contrast patterns from each pair of contrast datasets with each pattern bounded by a given gap constraint. The patterns mined from the datasets are used to build a supervised classifier for peptide folding prediction. Experimental results on two benchmark protein datasets and different learners assert the effectiveness of contrast patterns in capturing internal correlated amino acids for different types of folding.

References

Agrawal R., Imielinski T., and Swami A., 1993, *Mining association rules between sets of items in large databases*. In Proc. of SIGMOD.

Bowie J., Luthy R., Eisenberg D., 1991, A method to identify protein sequences that fold into a known three dimensional structure, *Science*, 253, pp.164-170.

Cristianini N., and Shawe-Taylor J., 2000. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

Cuff J., and Barton G. 2000, Application of multiple sequence alignment profiles to improve protein secondary structure prediction, *Proteins*, 40(3).

Ding C. and Dubchak I., 2001, Multi-class protein fold recognition using support vector machines and neural networks, *Bioinformatics*.

Domingos P., and Pazzani M., 1997, On the Optimality of the Simple Bayesian Classifier under Zero-One Loss, *Machine Learn*, 29.

Frishman D., and Argos P., 1997, Seventy-five percent accuracy in protein secondary structure prediction. *Proteins: Structure Function Genetic* 27, 329–335.

Ji X., Bailey J., and Dong G., 2005, Mining minimal distinguishing subsequence patterns with gap constraints, in *Proc. of the 5th International Conf. on Data Mining*.

Kabsch W. and Sander C., 1983, Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, *Biopolymers*. 22(12):2577-637.

Lesk A., 2002, *Introduction to Bioinformatics*, Oxford Univ Press

Ptitsyn O., and Finkelstein A., 1982, Theory of protein secondary structure and algorithm of its prediction. *Biopolymers*.

Qian N., and Sejnowski T., 1998, Predicting the secondary structure of globular proteins using neural network models, *Journal of Molecular Biology*, 202.

Quinlan J. 1993, *C4.5: Programs for machine learning*, Morgan Kaufmann Publishers, 1993.

Rost B. and Sander C. 1993. Prediction of protein secondary structure at better than 70% accuracy. *J. of Molecular Bio.*, 232.

Witten I. and Frank E., 2005, *Data mining: practical machine learning tools and techniques*, Morgan Kaufmann.

Zhu X. and Wu X., 2007, Mining complex patterns across sequences with gap requirements, in *Proc. of the 20th IJCAI Conf.*

Table 1: Peptide folding prediction results (Three classes)

Datasets	Folding classes	***Overall Accuracy***		***TPR***		***FPR***		***F-Measure***	
		CPM	SmpFld	CPM	SmpFld	CPM	SmpFld	CPM	SmpFld
RS126	H	0.709	0.307	0.426	0.169	0.027	0.114	0.548	0.198
	E			0.574	0.114	0.088	0.036	0.651	0.192
	C			0.893	0.422	0.421	0.234	0.775	0.511
CB513	H	0.669	0.281	0.475	0.189	0.07	0.121	0.543	0.223
	E			0.645	0.135	0.194	0.023	0.618	0.228
	C			0.743	0.404	0.24	0.157	0.735	0.518