AN ABSTRACT OF THE THESIS OF

Rongkun Shen for the degree of Master of Science in Computer Science presented on March 9, 2006.

Title: Protein Secondary Structure Prediction Using Conditional Random Fields and Profiles

Abstract approved:

_____

Thomas G. Dietterich

Protein secondary structure prediction plays a pivotal role in predicting protein folding in three-dimensions. Its task is to assign each residue one of the three secondary structure classes helix, strand, or random coil. This is an instance of the problem of sequential supervised learning in machine learning. This thesis describes a new model, TreeCRFpsi, for addressing this problem. TreeCRFpsi combines recent advances in machine learning with new sequence representations developed in molecular biology. The machine learning method, TreeCRF, constructs a conditional random field (CRF) by fitting a set of regression trees via an algorithm known as gradient tree boosting. The new sequence representation is the PSI-BLAST profile introduced by D. Jones, which is based on matching sequences of known protein structure against a much larger set of sequences drawn from the NCBI non-redundant protein sequence database. A new methodology of cross validation was developed and applied to choose the best parameter values for the model. The chosen parameters were the following: tree size of 10 leaves, sliding window size of 15 residues, and 3 rounds of PSI-BLAST searching. The mean three-state prediction accuracy reached 77.6% on both our new SD482 and the popular CB513 datasets. This result is the best among all published results. TreeCRFpsi improved especially on helix and strand predictions by 1-2.3 percentage points over the previous best methods. SOV99 scores were 74.6% and 73.9% for SD482 and CB513, respectively.

In addition, there was no apparent overfitting problem observed in our model. Besides achieving higher accuracy, TreeCRFpsi is the first secondary structure prediction method based on a well-defined probabilistic model, which makes it easier to use the output predictions as inputs to subsequent analysis steps.

Protein Secondary Structure Prediction
Using Conditional Random Fields and Profiles

by

Rongkun Shen

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented March 9, 2006

Commencement June 2006

Master of Science thesis of Rongkun Shen presented on March 9, 2006.

APPROVED:

Major Professor, representing Computer Science

Director of School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Rongkun Shen, Author

ACKNOWLEDGEMENTS

I would like to express my grateful thanks to my advisor, Thomas G. Dietterich, for his great ideas and extensive help during this research. Thanks to Thomas G. Dietterich and Adam J. Ashenfelter for the core TreeCRF codes and to Guohua Hao for his help on using the code.

I also would like to appreciate Christopher K. Mathews for his continuous support and considerateness.

I thank my wife Li Li, my lovely son Christopher, and my parents for their love, patience and support.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Since the late 1990s, the relative simplicity of DNA sequencing has led to the completion of several whole genome sequencing projects, such as the human genome project, the mouse genome project, the yeast genome project, the rice genome project, and so on. The number of available protein sequences is dramatically increasing accordingly. However, the number of protein structures is growing rather slowly due to the traditional strenuous and time-consuming approaches for structure determination, for example, X-ray crystallography and NMR. The gap between sequences and structures is rapidly widening (Table 1-1) (Hua and Sun, 2001). With the aid of computational power, protein structure prediction must play an important role in resolving the three-dimensional (3-D) protein folding problem. However, the direct *ab initio* prediction from protein sequence to three-dimensional structure is still very difficult. An alternative approach is to predict at an intermediate level, the secondary structures in one-dimension (1-D). It is widely believed that secondary structures can provide valuable information in determining a protein's three-dimensional structure.

Table 1-1. The gap between the number of protein sequences and the number of structures.

Database	Number of sequences	Number of residues
NCBI GenBank[a]	52,016,762	56,037,734,462[d]
Swiss-Prot[b]	205,780	74,898,419[e]
PDB[c]	31,629	---

Note: [a] Nucleic acid sequences, as of Dec. 15, 2005. [b] Protein sequences, as of Jan. 10, 2006. [c] Protein structures, as of Jan. 17, 2006. [d] Nucleotides. [e] Amino acid residues.

1.1 What Are Proteins?

While the nucleic acids store and transmit the genetic information of the cell, proteins are the products expressed from their corresponding genes in the genome. Protein functions are essential for life. They play various pivotal roles, such as storage and transportation, structural skeleton framework, muscle contraction, immune response, blood clotting, and the most important of all – enzymes – catalyzing a variety of reactions during life processes (Mathews *et al.,* 2000).

1.2 Protein Structures

Protein functions depend solely on their three-dimensional structures. There are four hierarchical levels of protein structure organization, which are the primary, secondary, tertiary, and quaternary structures.

1.2.1. Primary Structure

Proteins are composed of amino acid residues connected by covalent peptide bonds, which are planar and rigid (Figure 1-1). Protein primary structure is simply the amino acid sequence. It's linear, not branched and in one-dimension (Mathews *et al.,* 2000).

1.2.2. Secondary Structure

Because of the planar and rigid peptide pond plus the spatial restriction, the residues are not free to rotate and bend at all angles. Three basic local structures can be formed: α-helix (Figure 1-2), β-strand (Figure 1-3) and random coil (Mathews *et al.,* 2000). There are also some other secondary structures, such as the 3_{10}-helix, π-helix, isolated β-bridge,

turn and bend, but they are rare. The α-helix is a right-handed structure, with 3.6 amino acid residues and 13 backbone atoms per turn. Therefore it is also sometimes called the 3.6_{13} helix. The vertical distance between two neighboring turns is 5.4 Å. β-strands have a zigzag form and have two types: antiparallel (Figure 1-3(a)) and parallel (Figure 1-3(b)). In parallel strands, two sequence segments are in the same N-terminus to C-terminus direction; in antiparallel strands, they have opposite directions. Two sequence segments in remote positions in the sequence can form β-strands (Figure 1-3(c)). This is called a long-distance interaction, which is the most difficult part of secondary structure prediction.

NH_2-……-Gly-Tyr-Lys-Ala- ……-COOH

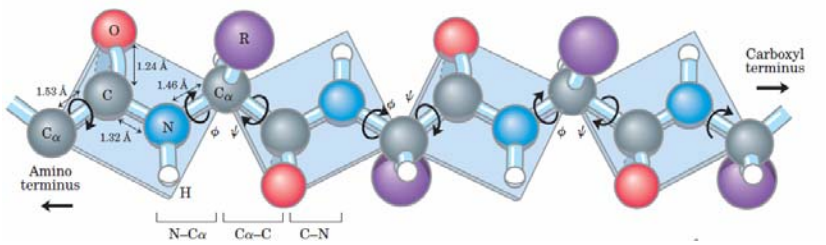


Figure 1-1. Primary structure of a partial protein sequence. (Courtesy of Nelson and Cox (2004))

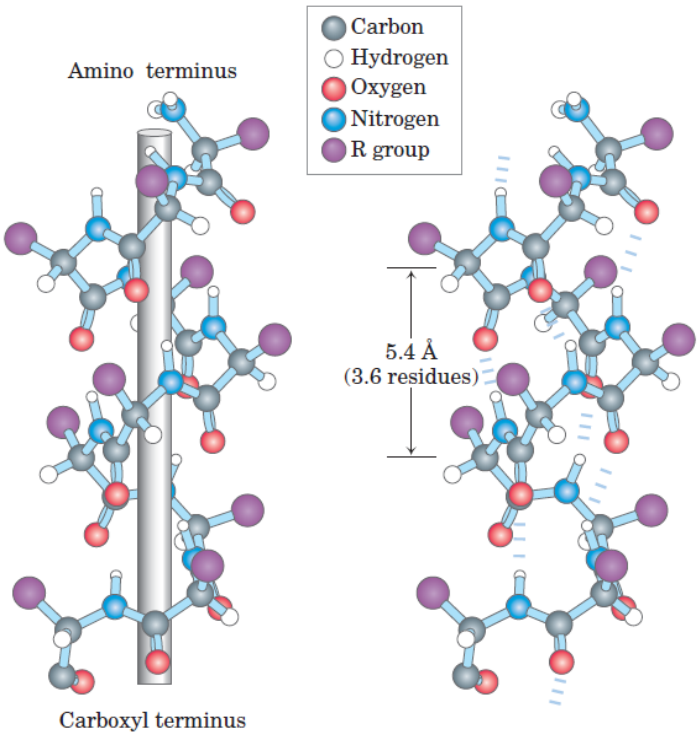


Figure 1-2. The structure of α-helix. (Courtesy of Nelson and Cox (2004))

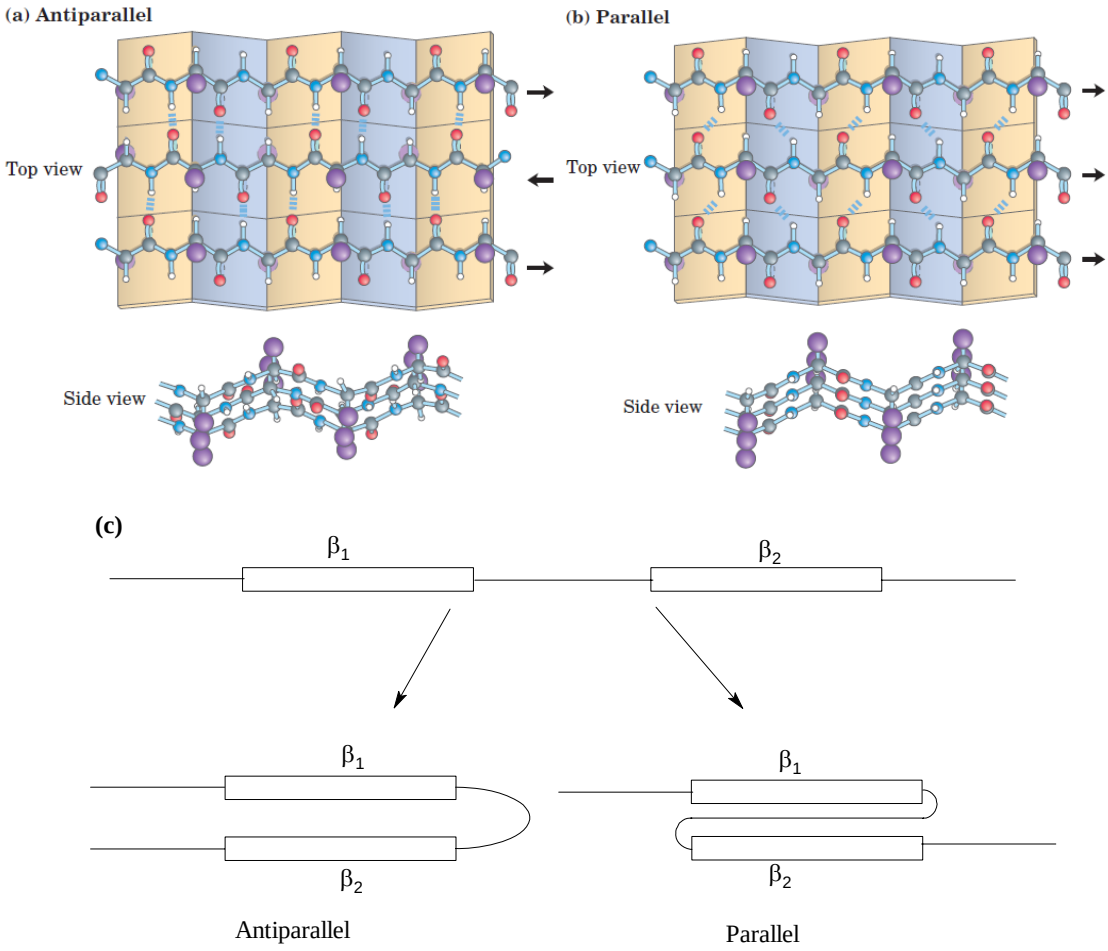
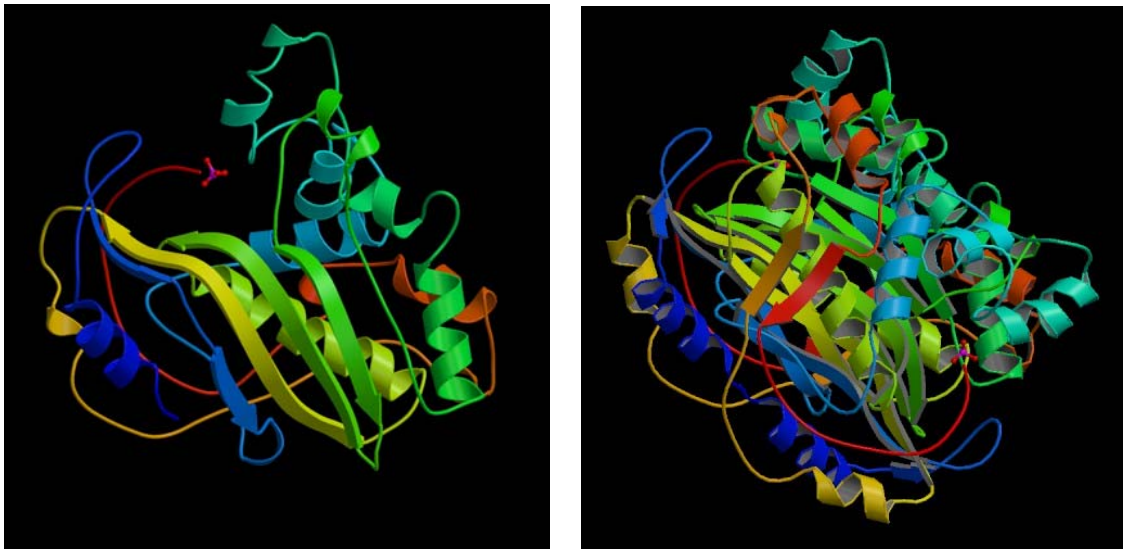


Figure 1-3. The structure of β-strands. (a) antiparallel β-strands; (b) parallel β-strands; (c) long-distance interaction between two β-strands. ((a) and (b), courtesy of Nelson and Cox (2004))

1.2.3. Tertiary structure

Protein tertiary structure is the folding in three-dimension (Mathews *et al.*, 2000). The regions consisting of secondary structures are folded into a specific compact structure for the entire polypeptide chain as exemplified in Figure 1-4(a).



(a) monomer (b) homodimer

Figure 1-4. An example of 3-D structure: bacteriophage T4 thymidylate synthase (PDB ID: 1TIS) (Finer-Moore *et al.,* 1994)

1.2.4. Quaternary structure

Some proteins are composed of two or more separate polypeptide chains, or subunits, which may be identical or different. The associations of these protein subunits in three-dimension complexes constitute the quaternary structure (Nelson and Cox, 2004). Figure 1-4(b) shows an example of quaternary structure, the homodimer (two identical chains) 3-D structure of phage T4 thymidylate synthase.

1.3 History of Protein Secondary Structure Prediction.

In 1951, Linus Pauling correctly proposed the conformation of helices and strands (Pauling and Corey, 1951; Pauling *et al.,* 1951). The first X-ray crystal structures of hemoglobin and myoglobin at atomic resolution were determined and published in 1960 (Kendrew *et al.,* 1960; Perutz *et al.,* 1960). In 1957, the first attempt to correlate some amino acids (e.g., proline) and α-helix structure was conducted by Szent-Guörgyi and colleagues (Szent-Guörgyi and Cohen, 1957).

1.3.1. The first generation

In the 1960s and 1970s, the first generation prediction methods were all based on single residue statistics, for example, the most famous work by Chou and Fasman (1974). The overall three-state (helix, strand and coil) average prediction accuracy was about 50%.

1.3.2. The second generation

The second generation between 1970s and 1990s combined a larger database of protein structures and statistics based on segments. The segments typically contain 11-21 consecutive residues from a protein sequence. The statistical methods were trying to assess the likelihood that the middle residue in the segment belongs to one of the three secondary structure classes. Almost all the algorithms available were applied, including neural networks (Qian and Sejnowski, 1988), graph theory (Mitchell *et al.*, 1992), and nearest-neighbor (Yi and Lander, 1993). The overall average prediction accuracy was slightly better than 60%.

1.3.3. The third generation

The first two generations had several problems. The three-state prediction accuracy was below 70% and, especially, the β-strand accuracy was below 50%, since long-range interactions could not be taken into consideration in all these algorithms and methods.

The third generation after the 1990s takes advantage of evolutionary information. Most of the mutations (i.e. exchange of one or several residues) do harm to a protein's stability and function. So residue substitution is unlikely to take place. However, the evolutionary pressure to keep the protein function determines that structure is more conserved than sequence (Lesk, 1991). All naturally evolved proteins containing more than 35% of

pairwise identical residues over 100 aligned residues have similar structures (Rost, 1998).

Combining larger databases with more advanced algorithms, the third generation prediction methods broke through the levels above 70% accuracy (Rost and Sander, 2000). These include PSIPRED (Jones, 1999) and PHDpsi (Przybylski and Rost, 2002) using multilayer neural networks, SVMpsi (Hua and Sun, 2001; Ward *et al.,* 2003; Kim and Park, 2003; Guo *et al.,* 2004) using support vector machines, SAM-T99sec (Karplus *et al.,* 1999) and HMMSTR (Bystroff and Shao, 2002) using hidden Markov models, and MEMMpsi using maximum entropy Markov models (Liu *et al.,* 2004), CRFpsi using conditional random fields (Liu *et al.,* 2004), and PSIMLR using multiple linear regression (Qin *et al.,* 2005).

1.4 PSI-BLAST

Evolutionary information is obtained from the powerful multiple sequence alignment tool called PSI-BLAST searching on a large protein sequence database. PSI-BLAST, or Position-Specific Iterated Basic Local Alignment Search Tool, uses the methods based on multiple sequence gapped alignment to search for similarities between protein query sequences and all the sequences in one or more protein databases. PSI-BLAST was first introduced by Altschul and colleagues at NCBI in 1997 (Altschul *et al.,* 1997) and improved by Schäffer *et al.* (2001).

PSI-BLAST uses position-specific scoring matrices (PSSMs) to score matches between query and database sequences, in contrast to BLAST which employs pre-defined scoring matrices such as BLOSUM62 (Altschul *et al.,* 1997). The BLOSUM matrix contains similarity scores for all possible substitutions of one amino acid with another during sequence alignment (Henikoff and Henikoff, 1992). BLOSUM62 employs a threshold of 62% identity or less and has become the standard for many alignment tools (Eddy, 2004). PSI-BLAST is a statistically driven search method that finds regions of similarity

between the query sequence and database sequences, and produces gapped alignments of those regions. PSI-BLAST may be more sensitive than BLAST, meaning that it might be able to find distantly related sequences that are missed in a BLAST search (Schäffer *et al*., 2001).

PSI-BLAST can repeatedly search the target databases, using a multiple alignment of high scoring sequences found in each search round to generate a new PSSM for use in the next round of searching. PSI-BLAST will iterate until no new sequences are found, or until the user specified maximum number of iterations is reached, whichever comes first. Normally, the first round of searching uses a standard scoring matrix, effectively performing a blastp (protein BLAST) search.

PSI-BLAST works as shown in Figure 1-5. The query sequence is first scanned for the presence of so-called low-complexity regions, that is, regions with a biased composition (e.g. transmembrane regions or coiled coils) likely leading to spurious hits (sequences in the database whose similarities exceed some specified value), which are excluded from alignment. Initially the program operates on a single query sequence by performing a gapped BLAST search against the database and finds significant local alignments (hits). The number of hits is controlled by the E-value threshold that the user specifies. E-value is the probability that a score or group of scores is observed as high as the observed score purely by chance searching against a database of this size. The smaller the E-value, the higher the similarity is. These local alignments are then used to construct a ‘multiple alignment’ and abstract a PSSM from this alignment. The program re-scans the database in a subsequent round using the PSSM generated in the previous round to find more homologous sequences. Iteration continues until the user decides to stop or the search converges.

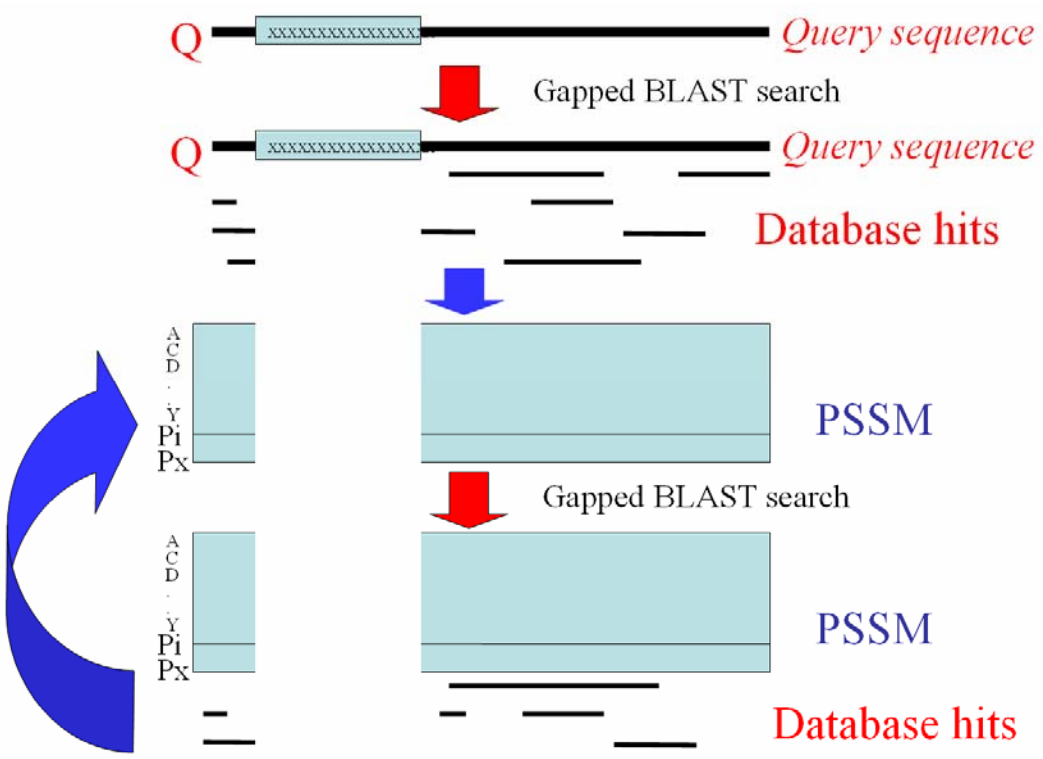


Figure 1-5. PSI-BLAST search and PSSM generation.

CHAPTER 2

CONDITIONAL RANDOM FIELDS VIA GRADIENT TREE BOOSTING

2.1 Sequential Supervised Learning

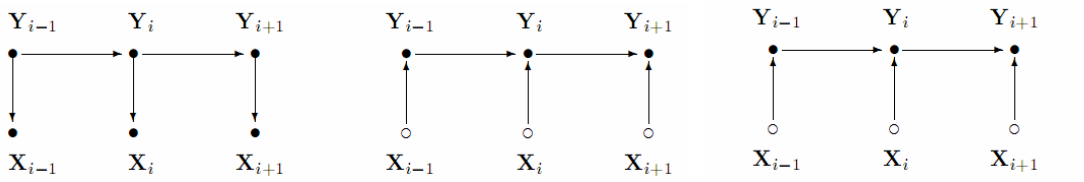
The goal of protein secondary structure prediction is to assign a secondary structure class - namely helix, strand or coil - to each amino acid residue in a protein sequence (Qian and Sejnowski, 1988). This is an instance of an abstracted prediction problem called the sequential supervised learning (SSL) problem. Sequential supervised learning (Dietterich, 2002; Dietterich *et al.,* 2004) is formalized as follows:

Given: A set of training examples of the form (X_i, Y_i), where each $X_i = (\mathrm{x}_{i,1},...,\mathrm{x}_{i,T_i})$ is a sequence of T_i feature vectors and each $Y_i = (y_{i,1},...,y_{i,T_i})$ is a corresponding sequence of class labels, $y_{i,t} \in \{1,...,K\}$

Find: A classifier H that, given a new sequence X of feature vectors, predicts the corresponding sequence of class labels $Y = H(X)$ accurately.

Many different methods have been applied to the protein secondary structure prediction problem. They can be generally grouped into traditional window-based models and graphical models. Traditional window-based models include neural networks (Qian & Sejnowski, 1988; Jones, 1999), K-nearest neighbors (Yi and Lander, 1993), and support vector machines (SVMs) (Hua and Sun, 2001; Ward *et al.,* 2003; Kim and Park, 2003; Guo *et al.,* 2004). The disadvantage of these window-based methods is that they only consider local information (Liu *et al.,* 2004).

Graphical models for sequential supervised learning mainly consist of hidden Markov models (HMMs) (Karplus *et al.*, 1999; Bystroff *et al.*, 2002), maximum entropy Markov models (MEMMs) (McCallum *et al.*, 2000) and conditional random fields (CRFs) (Lafferty *et al.*, 2001; Dietterich *et al.*, 2004).



(a) (b) (c)

Figure 2-1. Graphical models. (a) HMM, (b) MEMM, (c) CRF. (Modified from Lafferty *et al.* (2001) and Dietterich *et al.* (2004))

2.2 Hidden Markov Models

The hidden Markov models (HMMs) (Figure 2-1. (a)) are generative models that assume that the observations at time i in the sequence are generated according to the class at time i. The class label at time i is generated according to the previous class label, at time $i-1$. HMMs compute the joint distribution of observations X and states Y, $P(X,Y)$ and make predictions to compute the conditional distribution $P(Y \mid X)$ by applying Bayes rule. The model has to learn $P(y_i \mid y_{i-1})$ and $P(x_i \mid y_i)$. By the independence assumption, $P(x_i \mid y_i) = P(x_i \mid y_i, y_{i-1})$, and the joint probability $P(x_i, y_i \mid y_{i-1}) = P(x_i \mid y_i)P(y_i \mid y_{i-1})$. However, the independence assumption is also the drawback of the HMM, because it is hard to take account of overlapping long-distance interactions in the protein sequence.

2.3 Maximum Entropy Markov Models

The maximum entropy Markov models (MEMMs) and the conditional random fields (CRFs) are discriminative models shown in Figure 2-1, (b) and (c), respectively. The MEMM is formulated in terms of the conditional probabilities $P(y_i \mid y_{i-1}, X)$ based on an exponential model (McCallum *et al.*, 2000) as follows:

$$P(y_i \mid y_{i-1}, X) = \frac{1}{Z(y_{i-1}, X)} \exp\left[\sum_k \lambda_\kappa f_k (X, y_i, y_{i-1})\right],$$

where $Z(y_{i-1}, X)$ is a normalizing factor, f_k are features and λ_κ is the weight for feature f_k. MEMMs are better than HMMs in that MEMMs can include the long-distance interactions. However, they suffer from the so-called label bias problem due to the local normalization factor $Z(y_{i-1}, X)$. In short, the label bias means that the total probability received by y_{i-1} must be passed on to label y_i at time *i* even if x_i is completely incompatible with y_{i-1} (Lafferty *et al.*, 2001). For example, in Figure 2-2, we have a sequence of "rib" to pass through labels 1 and 2. After "r", both labels 1 and 2 have same probability. After "i", label 2 must still pass all of its probability forward, even though it was expecting "o". Therefore, both output strings "111" and "222" receive the same predicted probability.

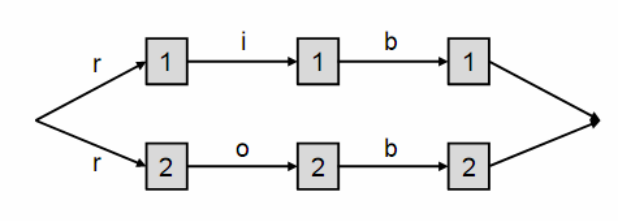


Figure 2-2. Label bias example.

2.4 Conditional Random Fields

Conditional random fields were first introduced by Lafferty and colleagues in 2001 (Lafferty et al., 2001). CRFs overcome the label bias problem of MEMMs by employing a global normalizing factor instead of an individual normalizer for each sequence item. The CRF computes the conditional probability $P(Y \mid X)$ based on an exponential model (Lafferty et al., 2001; Ashenfelter, 2003; Dietterich *et al.*, 2004):

$$P(Y \mid X) = \frac{1}{Z(X)} \exp\left[\sum_i \Psi_i \left(y_i, X\right) + \Psi_{i\text{-}1,i} \left(y_{i-1}, y_i, X\right) \right],$$

where $\Psi_i \left(y_i, X\right)$ and $\Psi_{i\text{-}1,i} \left(y_{i-1}, y_i, X\right)$ are potential functions representing how compatible y_i is with *X* and how compatible y_i is with a transition from y_{i-1} and with *X*. The exponential function ensures that $P(Y \mid X)$ is always positive, and the normalizing factor $Z(X) = \sum_{Y'} \exp\left[\sum_i \Psi_i \left(y_i', X\right) + \Psi_{i\text{-}1,i} \left(y_{i-1}', y_i', X\right) \right]$ ensures that the sum of all $P(Y \mid X)$ is equal to 1.

The two potential functions are represented by weighted combinations of binary features as Lafferty *et al* (2001) proposed:

$$\Psi \left(y_i, X\right) = \sum_a \beta_a g_a (y_i, X)$$

$$\Psi \left(y_{i-1}, y_i, X\right) = \sum_b \lambda_b f_b (y_{i-1}, y_i, X)$$

where β_a and λ_b are trainable weights, and g_a and f_b are Boolean functions.

Let $\Theta = \{\beta_1, \ldots, \lambda_1, \ldots\}$ be the parameters in the potential functions, then the objective function is to maximize

$$
\begin{aligned}
J(\Theta) &= \log \prod_j P(Y_j \mid X_j) \\
&= \sum_j \log \frac{1}{Z(X_j)} \exp\left[\sum_i \Psi_i \left(y_i, X\right) + \Psi_{i\text{-}1,i} \left(y_{i-1}, y_i, X\right) \right] \\
&= \sum_{j,i} \left[\Psi_i \left(y_{j,i}, X_j\right) + \Psi_{i\text{-}1,i} \left(y_{j,i-1}, y_{j,i}, X_j\right) - \log Z(X_j) \right] \\
&= \sum_{j,i} \left[\sum_a \beta_a g_a(y_{j,i}, X_j) + \sum_b \lambda_b f_b(y_{j,i-1}, y_{j,i}, X_j) - \log Z(X_j) \right].
\end{aligned}
$$

where j indexes the training sequences.

2.5 CRF Training via Gradient Tree Boosting

Lafferty *et al.* (2001) studied how to train CRFs via an iterative scaling algorithm. However, they reported that it was very slow. Dietterich *et al.* (2001) studied the gradient descent algorithms and it turned out to be very slow too. Wallach (2003) applied L-BFGS for CRF training and found that it was significantly faster. In 2004, Dietterich's group reported a much faster algorithm, called gradient tree boosting (Ashenfelter, 2003; Dietterich *et al.*, 2004).

The implementation of TreeCRF employs both gradient descent and boosted regression trees (Dietterich *et al.*, 2004). The potential functions $\Psi\left(y_i, X\right)$ and $\Psi\left(y_{i-1}, y_i, X\right)$ are represented as weighted sums of regression trees. Let

$$F^{y_i}\left(y_{i-1}, X\right) = \Psi\left(y_i, X\right) + \Psi\left(y_{i-1}, y_i, X\right)$$

be a function computing the sum of potentials of label y_i given values for label y_{i-1} and the input features X. Then the CRF becomes

$$P(Y \mid X) = \frac{1}{Z(X)} \exp \sum_i F^{y_i}\left(y_{i-1}, X\right).$$

The functional gradient of $\log P(Y \mid X)$ with respect to $F^{y_i}\left(y_{i-1}, X\right)$ is

$$\frac{\partial \log P(Y \mid X)}{\partial F^{v}\left(u, w_d(X)\right)} = I(y_{d-1} = u, y_d = v) - P(y_{d-1} = u, y_d = v \mid w_d(X)),$$

where $w_d(X)$ is a window in the sequence X centered at x_d; $I(y_{d-1} = u, y_d = v)$ is 1 if the transition $u \to v$ is observed from position $d-1$ to position d in the sequence Y and 0 otherwise; $P(y_{d-1} = u, y_d = v \mid w_d(X))$ is the predicted probability of this transition according to the current potential functions; each window $w_d(X)$ is assumed unique.

The normalizing factor, $Z(X)$, is computed by the forward-backward algorithm. The forward recursion is defined by

$$\alpha(k,1) = \exp F^{k}(\perp, w_1(X))$$

$$\alpha(k,i) = \sum_{k'}\left[\exp F^{k}(k', w_i(X))\right] \cdot \alpha(k', i-1),$$

where y_t is $\perp$ for $i < 1$. The backward recursion is

$$\beta(k,T) = 1$$

$$\alpha(k,i) = \sum_{k'}\left[\exp F^{k'}(k, w_{i+1}(X))\right] \cdot \beta(k', i+1).$$

The variables k and k' iterate over the possible class labels. Finally the normalizer $Z(X)$ can be computed at position i as

$$Z(X) = \sum_{k} \alpha(k,i)\beta(k,i).$$

The size of the regression trees is controlled by setting a limit on the number of allowed leaf nodes. There is a tradeoff between the expressive power and the learning speed in choosing the tree size (Ashenfelter, 2003). A CRF with a large tree size is more expressive and learns faster, but has the risk of overfitting. A CRF with a small tree size generalizes better to the test data, since each tree is less expressive.

The regression tree CRF can make more accurate predictions, and prevent overfitting due to the “ensemble effect” of combining the regression trees. The most important advantage of the TreeCRF algorithm is that it speeds up the computation significantly compared to the algorithms mentioned above (Ashenfelter, 2003; Dietterich *et al.*, 2004).

CHAPTER 3

MATERIALS AND METHODS

3.1 The CB513 Dataset and Generation of our SD482 Dataset

We employed 2 datasets in this research: CB513 and SD482. Many papers have reported results obtained from the CB513 dataset using different models and approaches. The CB513 dataset contains proteins that are non redundant and does not contain any membrane proteins, so it is suitable for training and testing new secondary structure prediction methods (Cuff and Barton, 1999). Because membrane proteins have quite different amino acid compositions and structures from globular proteins, in this work we only deal with soluble globular proteins. The CB513 dataset can be downloaded from the http://www.compbio.dundee.ac.uk.

The SD482 dataset is a subset of CB513 constructed as follows. At first, the whole CB513 dataset was screened and the 16 sequences having fewer than 30 residues were removed since they lack well-defined secondary structures (Cuff and Barton, 1999). During the first iteration of PSI-BLAST, 15 sequences returned fewer than 12 hits (sequences) in the latest NCBI non-redundant protein sequence database (see Section 3.4). These 15 sequences were removed too, because they do not generate useful PSI-BLAST alignment information, and this further deteriorates the predictions. Finally, the remaining 482 sequences form our SD482 dataset. These steps are similar to those of Cuff and Barton (2000) and Kim and Park (2003).

3.2 Cross-Validation: Traditional and New Methodology

Cross-validation is one of several approaches for examining how well the model learned from the training data will predict on unseen testing data. It is also one of the most popular techniques for detecting overfitting, and it can be applied to any learning algorithm. *K*-fold cross validation divides the data set into *k* subsets, and the holdout method is run *k* times. Each time, one of the *k* subsets is used as the testing set and the remaining (*k*-1) subsets are put together to form a training set. Figure 3-1(a) shows a 10-fold cross validation. Then, the results from all *k* trials are averaged (Russell and Norvig, 2003). In order to fairly compare our TreeCRFpsi model with others, we applied 7-fold cross-validation on CB513 dataset and SD482 as well, as some previous papers did (Jones, 1999; Kim and Park, 2003; Liu *et al.*, 2004).

The old method (Figure 3-1(a)) can be represented as follows:

a. Randomly divide the data into K subsets $\mathrm{F}_1, \ldots, \mathrm{F}_\mathrm{K}$
b. For each parameter value θ
    For each $k = 1 \text{ to } \mathrm{K}$ do
        construct the training set = union of $\mathrm{F}_1, \ldots, \mathrm{F}_\mathrm{K}$ except F_k
        construct the test set = F_k
        train on the training set
        test on the test set
    Compute the combined test set error (total errors / total predictions)
    Remember the value θ^* that gives the best combined test error.
c. Report the best combined test error.

The problem is that the best parameter value θ^* is based on test set performance. This requires $\mathrm{W}*\mathrm{K}$ runs of the learning algorithm, where W = number of parameter values and K = number of folds.

Many authors use cross-validation to choose parameter values that control the learning algorithm, such as the regression tree size and the number of training iterations. However, this can lead to test set contamination unless this traditional cross-validation method is modified. The following pseudo code describes our new method, which is also shown in Figure 3-1(b).

a. Randomly choose the development set D
b. divide remaining data at random into K subsets $\mathrm{F}_1,...,\mathrm{F}_\mathrm{K}$
c. For each $k=1 \text{ to } \mathrm{K}$ do
   construct the training set = union of $\mathrm{F}_1,...,\mathrm{F}_\mathrm{K}$ except F_k
   construct the test set = F_k
   For each candidate parameter value θ
      train on the training set using θ
      test on the development set D
   Let θ_k be the best parameter setting found
   Train using θ_k on the training set and compute predictions on the test set. Save the predictions for later
d. Compute the performance measures for the K-fold cross validation by combining the predictions saved from each fold.

This also requires W*K runs (where W = number of parameter values and K = number of folds), so the amount of CPU time is just as good without test set contamination problem.

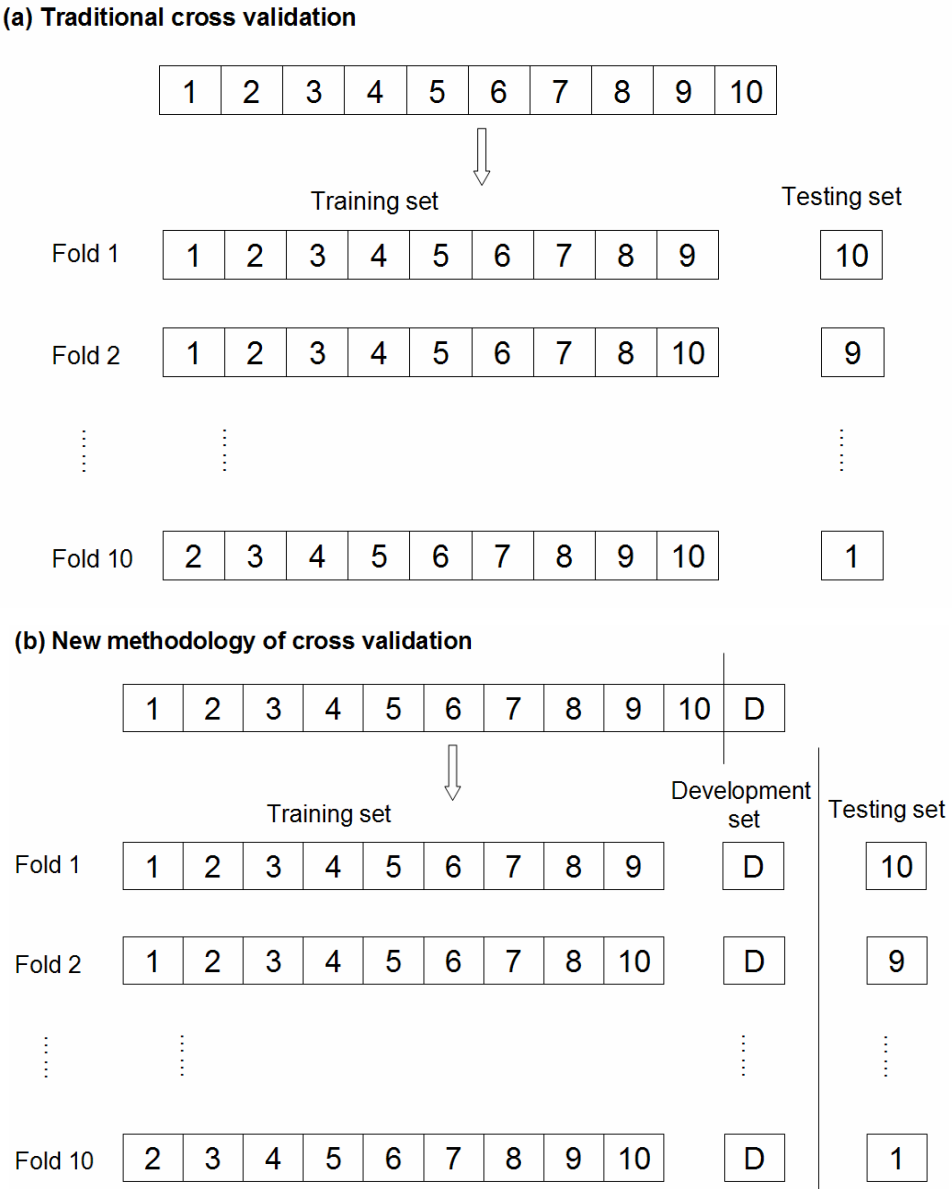


Figure 3-1. Cross validation methods: (a) traditional; (b) new methodology.

In our new methodology (Figure 3-1 (b)), we applied the training set against the development set to choose the optimal values and the testing set was never touched. First, take out ~10% (42 sequences) of the whole dataset SD482 to be the development set. Second, divide the remaining 440 sequences evenly into 10 parts for 10-fold cross-validation with 44 sequences in each part. Third, on each fold, run the 396 sequences as the training set against the development set (42 sequences) to select the

tuning parameter values for the model. These parameters include the number of leaves for the TreeCRF, window size, the number of iterations and etc. Fourth, use the selected parameters to run the 396 sequences as the training set against the testing set (44 sequences each). Finally, compute the overall performance from all ten folds (Shen and Dietterich, 2006).

3.3 Reduction of Secondary Structure Classes

The CB513 dataset adopts DSSP to represent the secondary structure classes. DSSP is an acronym for Definition of Secondary Structure of Proteins, which is based on hydrogen bonding patterns and geometrical constraints (Kabsch and Sander, 1983). DSSP has eight secondary structure classes: H(α-helix) and G(3_{10}-helix), I(π-helix), E(β-strand) and B(isolated β-bridge), T(turn), S(bend) and _ (other). We reduced these 8 classes to 3 classes (**H, E, C**) according to the scheme defined by Rost and Sander (Rost and Sander, 1993):

H, G to **H**; E, B to **E**; and the remaining to **C**.

Most papers employ this same reduction (Jones, 1999; Cuff and Barton, 1999; Cuff and Barton, 2000; Hua and Sun, 2001; Kim and Park, 2003; Liu *et al.*, 2004). Different reduction methods can change the measured prediction accuracy (Cuff and Barton, 2000). For example, the following scheme,

H to **H**; E to **E**; and the remaining to **C**,

would simply increase the prediction accuracy since helices and strands are the bottleneck of prediction (Petersen *et al.*, 2000; Kim and Park, 2003).

3.4 Generation of PSSM Raw Profiles

The position-specific scoring matrix (PSSM) raw profiles were generated by PSI-BLAST searching against the NCBI non-redundant protein sequence database. The PSSM raw profile has $20 \times N$ elements, where N represents the length of the sequence and each element represents the log-likelihood of a particular residue being substituted by others.

A large non-redundant protein sequence database from NCBI, downloaded from ftp://ftp.ncbi.nih.gov/blast/db, was employed as the searching database for PSI-BLAST. The latest database contains 2,354,365 sequences and 800,120,167 total residues with entries from GenPept, Swissprot, PIR, PDF, PDB, and NCBI RefSeq. Before running PSI-BLAST, the database itself was filtered by PFILT from PSIPRED (Jones, 1999) to remove low-complexity regions (i.e., transmembrane regions and coiled-coil segments).

The sequences in the two datasets were then searched by PSI-BLAST against the NCBI non-redundant protein sequence database to generate PSSM raw profiles. The NCBI toolkit can be obtained from ftp://ftp.ncbi.nih.gov, and the PSI-BLAST executables can be downloaded from ftp://ftp.ncbi.nih.gov/blast. Different numbers of iterations were executed and evaluated. Details will be discussed below.

3.5 The Set of Features for Feeding TreeCRF

The input to TreeCRF is a sliding window from position $x_{i-\left\lfloor \frac{w}{2} \right\rfloor}$ to $x_{i+\left\lfloor \frac{w}{2} \right\rfloor}$ for window size of w (details in Section 3.7). The set of features of x_i contains the PSSM profiles (20 columns) and the secondary structure class (1 column). Figure 3-2 shows detailed examples of features.

3.6 Transformation from Raw PSSM Profiles to Thermometer Representation

The raw PSSM profiles generated from PSI-BLAST are further transformed into the format as shown in Figure 3-2. The first column is the sequence number, the second, the residue number, 3rd-22nd, the PSSM raw profiles (20 columns), and the last, the secondary structure classes. All scores in the raw PSSM profiles range from -8 to 13. Each score was then converted into a thermometer representation:

For a given range of $[a,b]$, a and b are integers, a particular number c $(a \le c \le b)$ can be represented as all 1's between a and c and all 0's between c and b, totally $b-a+1$ columns.

For example, the thermometer representation "-5" in [-8, 13] is

`1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0,` totally 22 columns;

or "8" is

`1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0.`

The reason for this transformation is that the TreeCRF code is optimized for working with Boolean features, and this allows the regression tree to employ tests of the form $s > \theta$, where s is the PSSM profile score and θ is a constant threshold value.

3.7 Transformation to Sparse Representation, Windowization and Feeding to TreeCRF

The WINDOWIZE program converts the basic sequence of residues into different sizes of sliding window. For example, for a window size of 3, given a sequence of observations $\{x_1, x_2, x_3, \ldots, x_n\}$ and a sequence of class labels $\{y_1, y_2, y_3, \ldots, y_n\}$, the sliding windows would be $\left(\langle B, x_1, x_2\rangle, y_1\right)$, $\left(\langle x_1, x_2, x_3\rangle, y_2\right)$, $\left(\langle x_2, x_3, x_4\rangle, y_3\right)$, …, $\left(\langle x_{n-1}, x_n, B\rangle, y_n\right)$, where *B* is blank of null value (Ashenfelter, 2003). The profiles are then fed into our TreeCRF program. TreeCRF outputs the trained CRFs, predicted secondary structure classes (helix, strand, coil) and the three-state overall percentage of accuracy - Q_3.

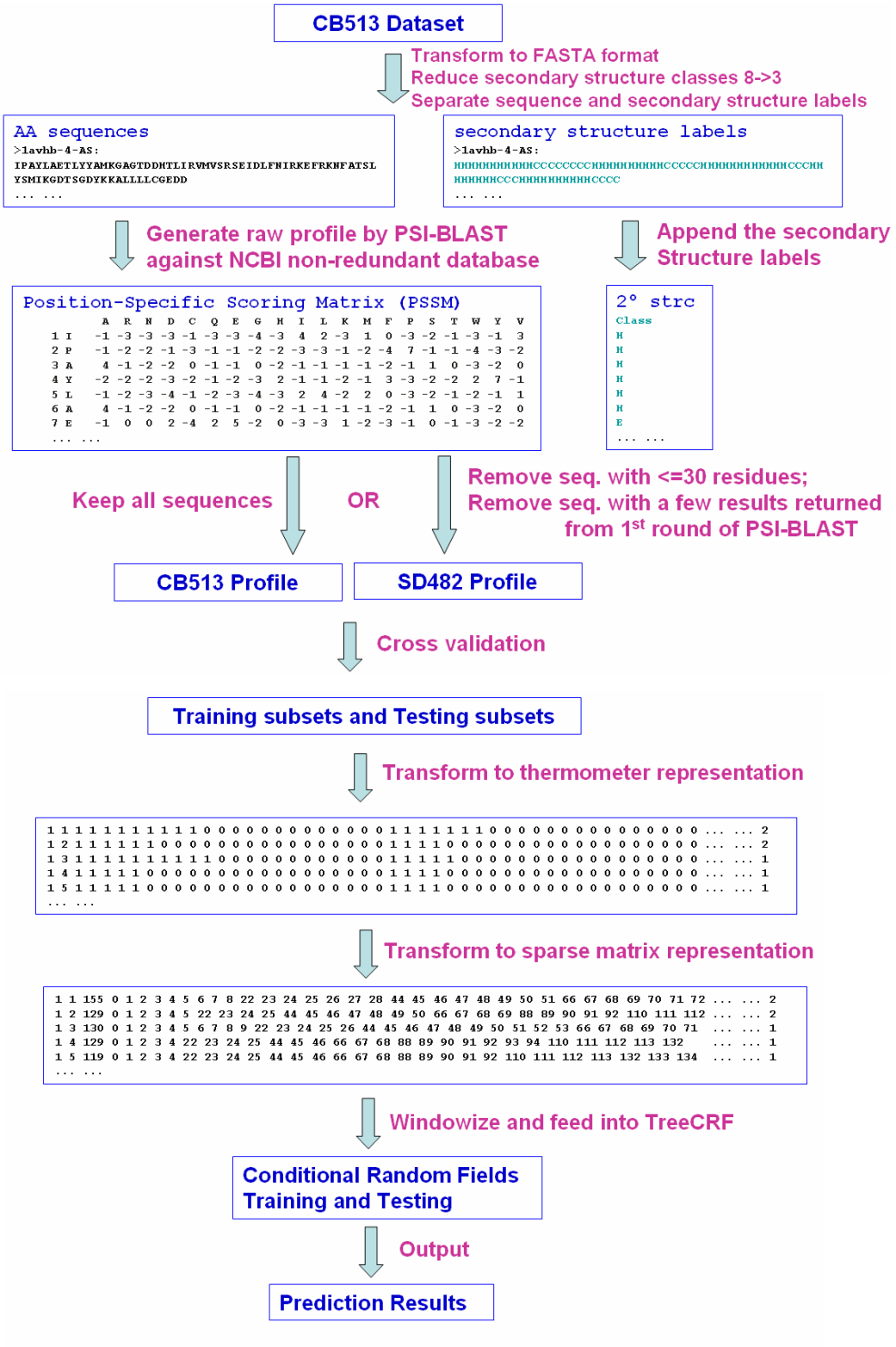


Figure 3-2. A flowchart of the TreeCRFpsi experiments.

3.8 Assessment of Prediction Accuracy

Several standard evaluation methods were employed to measure the secondary structure prediction accuracy. The first, called Q_3, measures the overall three-state percentage of correctly predicted residues, as follows,

$$Q_3 = \frac{\sum_{i\in\{H,E,C\}} \text{number of residues correctly predicted in state } i}{\sum_{i\in\{H,E,C\}} \text{number of residues observed in state } i} \times 100,$$

where conformation state i is H (helix), E (strand) or C (coil).

Second, the per residue accuracy Q_i for each type i of secondary structure was calculated as

$$Q_i = \frac{\text{number of residues correctly predicted in state } i}{\text{number of residues observed in state } i} \times 100,$$

where conformation state i is H (helix), E (strand) or C (coil).

The third measure is the segment overlap measure (SOV). It calculates secondary structure segments (strings of identical states) instead of individual residues, and it is more structurally meaningful. SOV was proposed by Rost *et al*. (1994) and re-defined by Zemla *et al*. (1999) (Figure 3-3). SOV is calculated as

$$SOV = \left[\frac{1}{N} \sum_{i\in\{H,E,C\}} \sum_{S(i)} \frac{\text{minov}(s_1,s_2) + \delta(s_1,s_2)}{\text{minov}(s_1,s_2)} \times len(s_1) \right] \times 100,$$

where s_1 is the observed segment, s_2 is the predicted segment to be evaluated, $S(i)$ is the set of all overlapping pairs of segments (s_1, s_2) in secondary structure class i, $len(s_1)$ is the number of residues in segment s_1, $\text{minov}(s_1,s_2)$ is the length of the actual overlap and $\text{maxov}(s_1,s_2)$ is the total extent of the segment. The normalization factor $N = \sum_{i\in\{H,E,C\}} N_i$ is the sum of all three states (i = helix, strand, or coil) and

$\delta(s_1, s_2)$ is defined as

$$\delta(s_1, s_2) = \min \begin{Bmatrix} (\mathrm{maxov}(s_1, s_2) - \mathrm{minov}(s_1, s_2)); \\ \mathrm{minov}(s_1, s_2); \\ \mathrm{int}(len(s_1)/2); \\ \mathrm{int}(len(s_2)/2) \end{Bmatrix},$$

where $\min\{x_1; x_2; x_3; \ldots; x_n;\}$ is the minimum of n integers.

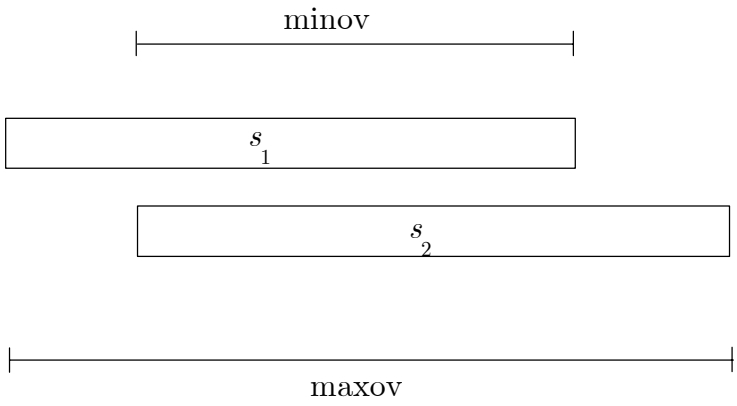


Figure 3-3. Segment of overlaps (SOV).

In our experiments, Q_3, Q_i and SOV scores were calculated by the SOV program provided by Zemla *et al.* (1999), which is available in http://predictioncenter.org/local/sov/sov.html.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Effect of Different Numbers of PSI-BLAST Iterations

PSI-BLAST can do BLAST searches iteratively until the user stops it as described in Chapter 1. We ran different numbers of iterations of PSI-BLAST in order to find out what number of rounds is the best for secondary structure prediction. Figure 4-1 shows the three-state prediction results employing the profiles generated from 2, 3, 4, 5, 6, 8, and 16 rounds of PSI-BLAST iterations, assayed on the same training and testing sub-datasets from our SD482 dataset. One-third of the sequences in SD482 were randomly assigned into the testing sub-dataset, and the remaining two-thirds went into the training sub-dataset. We found that three rounds gave the highest score, which is consistent with Jones' report (Jones, 1999). The more rounds of PSI-BLAST iterations, the worse the results were. In the following experiments, we just adopted the PSSM profiles from 3 rounds of PSI-BLAST iterations.

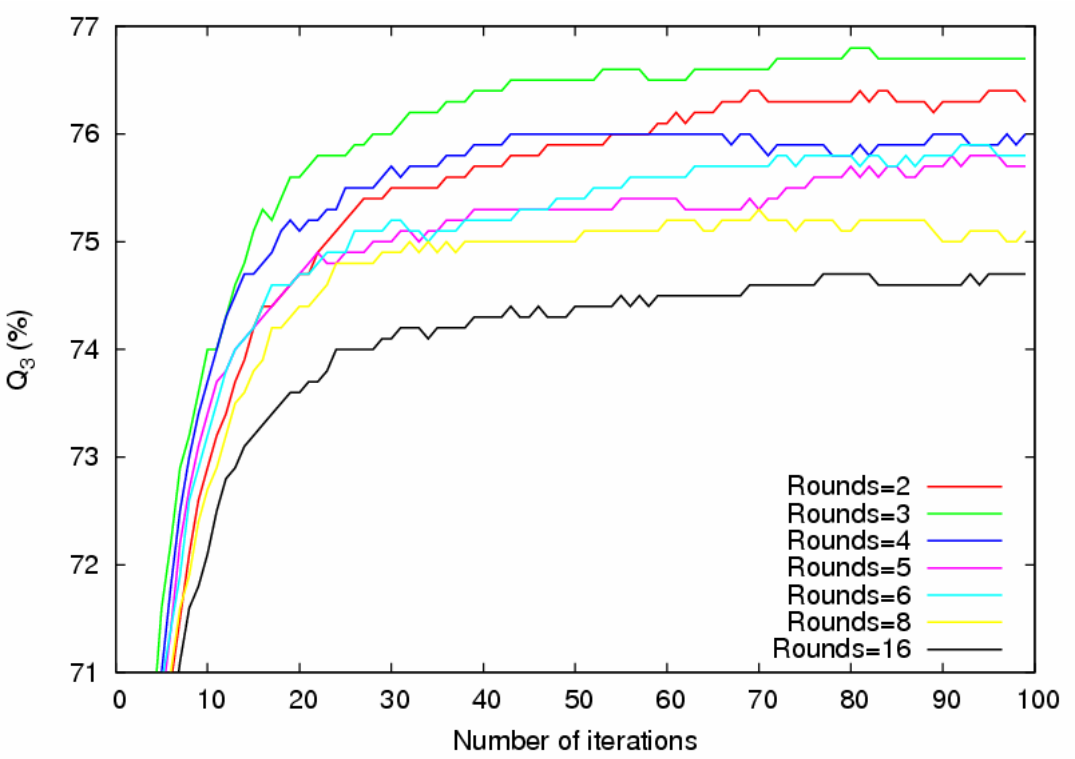


Figure 4-1. Prediction comparisons on different rounds of PSI-BLAST iterations. Assays were based on the same training and testing sub-datasets from SD482 dataset, where training sub-dataset contains 321 sequences and testing, 161 sequences.

4.2 A New Methodology of Cross Validation to Choose the Best Parameter Values

Our TreeCRF model has several parameters that can be adjusted to achieve the best performance. First, the number of leaves controls how many leaves are allowed in each CRF regression tree. Second, the window size is the size of the sliding window. If the window size is too short, it might miss some important classification information (Kim and Park, 2003). However, if the windows size is too long, it might suffer from inclusion of unnecessary noise. It can also exhaust memory of the computer. In other words, if the sliding window size is not appropriate, the signal-to-noise ratio will be low (Hua and Sun, 2001). Third, the number of iterations determines when to stop the training.

As indicated in Chapter 3, the traditional cross validation has the problem of test set contamination for choosing optimal parameter values. The new cross-validation methodology was applied to determine the optimal parameter values using the training set against the development set. In each fold, the test set was never touched so the contamination problem was avoided. About 10% (42 sequences) of SD482 was randomly selected as the development set. The remaining 440 sequences were divided via the 10-fold cross validation method. In each fold, we fed the TreeCRF program with the training and the development sets. We applied different number of leaves (i.e., 10, 20, 40, 60, 80 and 160), different window sizes (i.e., 3, 5, 7, 9, 11, 13, 15, 17, 19 and 21) and a fixed iteration number (i.e., 200). Table 4-1 shows the best window size and the best number of leaves obtained for each fold, and the best Q_3 values at the particular number of iterations as well. From the table, the overall best number of leaves was 10 (20 in several folds). Although we were able to choose fewer leaves and hence increase speed (e.g. 5 or 8), we employed 10 leaves since a TreeCRF model with fewer than 10 leaves might lose the expressive power of feature combination in the regression trees. Table 4-1 also gives us the overall best window size of 15 (13 in several folds). We plotted the averaged data from 10 folds for the purpose of visualization. Figure 4-2 also shows that the best number of leaves was 10 and Figure 4-3 gives the best window size of 15. We were not able to provide the results for W=19 and 21, because the TreeCRF program overflowed memory while running. The sliding window size of 15 is consistent with that of other published papers (Jones, 1999; Kim and Park, 2003; Liu *et al.*, 2004) and very close to W=13 in Hua and Sun (2001) and Qian and Sejnowski (1988).

Table 4-1. Determination of optimal parameter values on each fold in new cross validation method on SD482. Results were obtained from each fold's training set and the common development set.

Fold	Best values			
	Number of leaves	Window size	Number of iterations	Q_3(%)
1	10	15	191	76.1
2	10	13	111	75.7
3	20	15	165	76.0
4	10	15	192	75.5
5	10	15	193	75.7
6	10	15	92	75.9
7	10	15	125	76.0
8	20	13	139	76.3
9	20	15	119	75.5
10	10	15	161	75.6

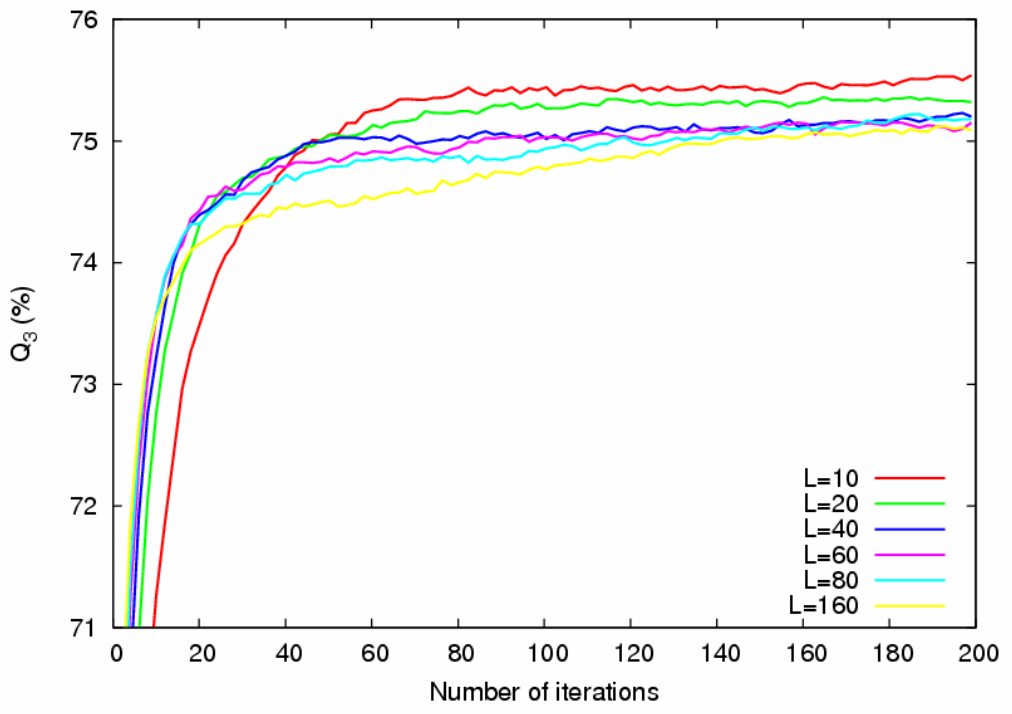


Figure 4-2. Prediction comparisons on different number of leaves of the regression trees. All results were computed on the development set.

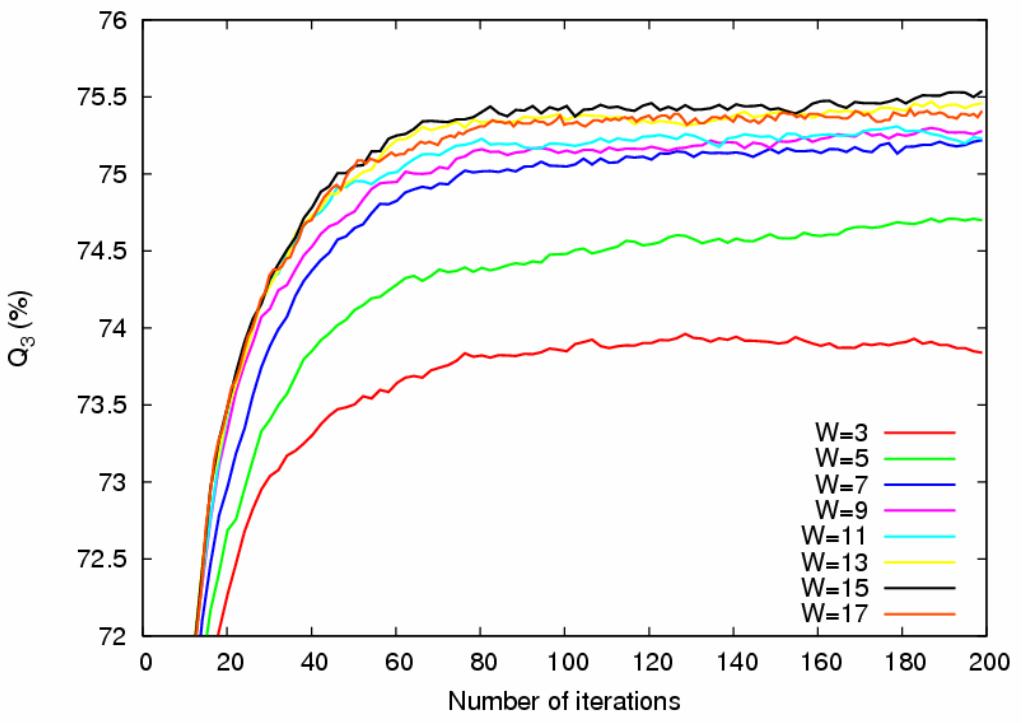


Figure 4-3. Prediction comparisons on different window sizes of input. All results were computed on the development set.

4.3 Results from New Cross Validation Methodology on SD482 Dataset

Once the optimal number of leaves and the sliding window size had been determined, we conducted the prediction on the training and testing sub-datasets in each fold for the new cross validation methodology on SD482. The three-state prediction accuracy (Q_3), helix accuracy (Q_H), and strand accuracy (Q_E) achieved 77.3%, 80.2%, and 65.7%, respectively (Table 4-2). SOV99 was 74.5%. All the results were obtained at the 100th iteration of TreeCRF fitting. Because there was no apparent overfitting observed during the experiments (Figures 4-1, 4-2 and 4-3), better prediction results would be expected by extending the training to larger number of iterations (e.g., 400).

Table 4-2. Results of new cross validation methodology on SD482. Results were obtained at the 100th iteration, 10 leaves and the window size of 15.

Fold	Q_3	Q_H	Q_E	Q_C	SOV99
1	77.3	80.9	66.7	78.7	74.3
2	75.1	78.6	63.4	79.3	74.5
3	78.1	78.4	69.2	82.8	74.6
4	78.1	83.5	63.4	81.6	75.6
5	77.4	79.4	67.5	81.3	73.6
6	77.3	82.6	64.7	79.9	72.7
7	76.9	81.0	63.7	79.6	74.9
8	77.0	77.5	65.6	82.1	75.0
9	78.7	80.1	69.1	83.1	76.4
10	77.3	80.3	64.8	81.4	73.3
Average	**77.3**	**80.2**	**65.8**	**81.0**	**74.5**

4.4 Results from Traditional Cross Validation on SD482 and CB513 Datasets

The traditional cross validation method is dangerous. However, we wanted to compare how well TreeCRF performed to other published studies. Tables 4-3 and 4-4 show the average prediction results from SD482 and CB513 datasets, respectively. Q_3, Q_H and Q_C results on both datasets reached 77.6%, ~80.6% and ~81.2%, respectively. However, for Q_E and SOV99, SD482 had better prediction accuracy than CB513. We suggest that the exclusion of short sequences (< 30 residues) and sequences having few results during the first round of PSI-BLAST search did help promote the prediction on Q_E and SOV99, but not on Q_3, Q_H and Q_C. Further analysis on some of these omitted sequences will be given in the next section. Overfitting did not occur for both datasets even we extended the

training to 400 iterations (Figure 4-4).

Table 4-3. Results of traditional 7-fold cross validation on SD482. Results were obtained from the 351st iteration, L=10, W=15.

Fold	Q_3	Q_H	Q_E	Q_C	SOV99
1	76.7	76.3	65.1	82.5	73.2
2	78.3	81.8	67.1	81.0	75.0
3	76.6	79.3	66.3	80.1	72.6
4	77.7	83.2	65.2	79.9	78.2
5	78.2	80.0	68.9	82.6	76.5
6	79.0	84.3	66.0	80.6	73.7
7	76.9	78.4	67.3	80.5	73.1
Average	**77.6**	**80.5**	**66.6**	**81.0**	**74.6**

Table 4-4. Results of traditional 7-fold cross validation on CB513. Results were obtained from the 346th iteration, L=10, W=15.

Fold	Q_3	Q_H	Q_E	Q_C	SOV99
1	77.3	79.4	67.5	81.1	71.9
2	77.0	80.7	63.5	80.7	73.3
3	76.0	76.3	65.0	82.4	70.6
4	77.8	80.5	66.7	81.4	74.1
5	77.9	82.3	66.5	80.9	75.4
6	78.4	83.2	65.9	80.5	75.3
7	79.1	82.0	67.7	81.7	76.9
Average	**77.6**	**80.6**	**66.1**	**81.2**	**73.9**

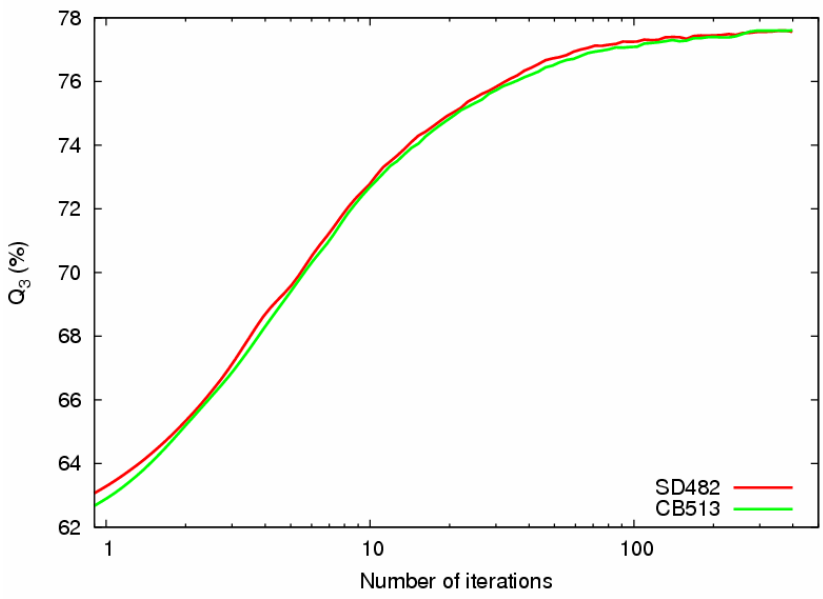


Figure 4-4. Averaged three-state predictions on SD482 and CB513 datasets through 400 iterations. Results were based on 7-fold cross validation at L=10, W=15.

4.5 Assessment of Predictions per Sequence on SD482 and CB513 datasets

Figure 4-5 plots histograms showing the pattern of predictions per protein sequence fitted by Gaussian distribution peaking at 80-83.3% for both SD482 and CB513 datasets. The means and variances were 80.9% and 7.0 for SD482 dataset; 80.3% and 7.2 for CB513 dataset. We found that 74.7% and 72.3% of the sequences were correctly predicted at over 76.6% (close to the average of 77.6%) for SD482 and CB513 datasets, respectively (Table 4-5). 89.0% and 88.9% of sequences were correctly predicted between 70.0% and 93.3% for SD482 and CB513, respectively. Generally speaking, the performance on SD482 was slightly better than that on CB513. Careful analysis of the sequences with Q_3 of less than 53.3% revealed that there were 12 sequences in the CB513 dataset, but only 7 such sequences in the SD482, among which 5 sequences were identical. Five over the remaining 7 sequences in CB513 contain fewer than 30 amino acid residues. We suggest that these short sequences do not possess well-defined secondary structures and their PSSM profiles contain poor information.

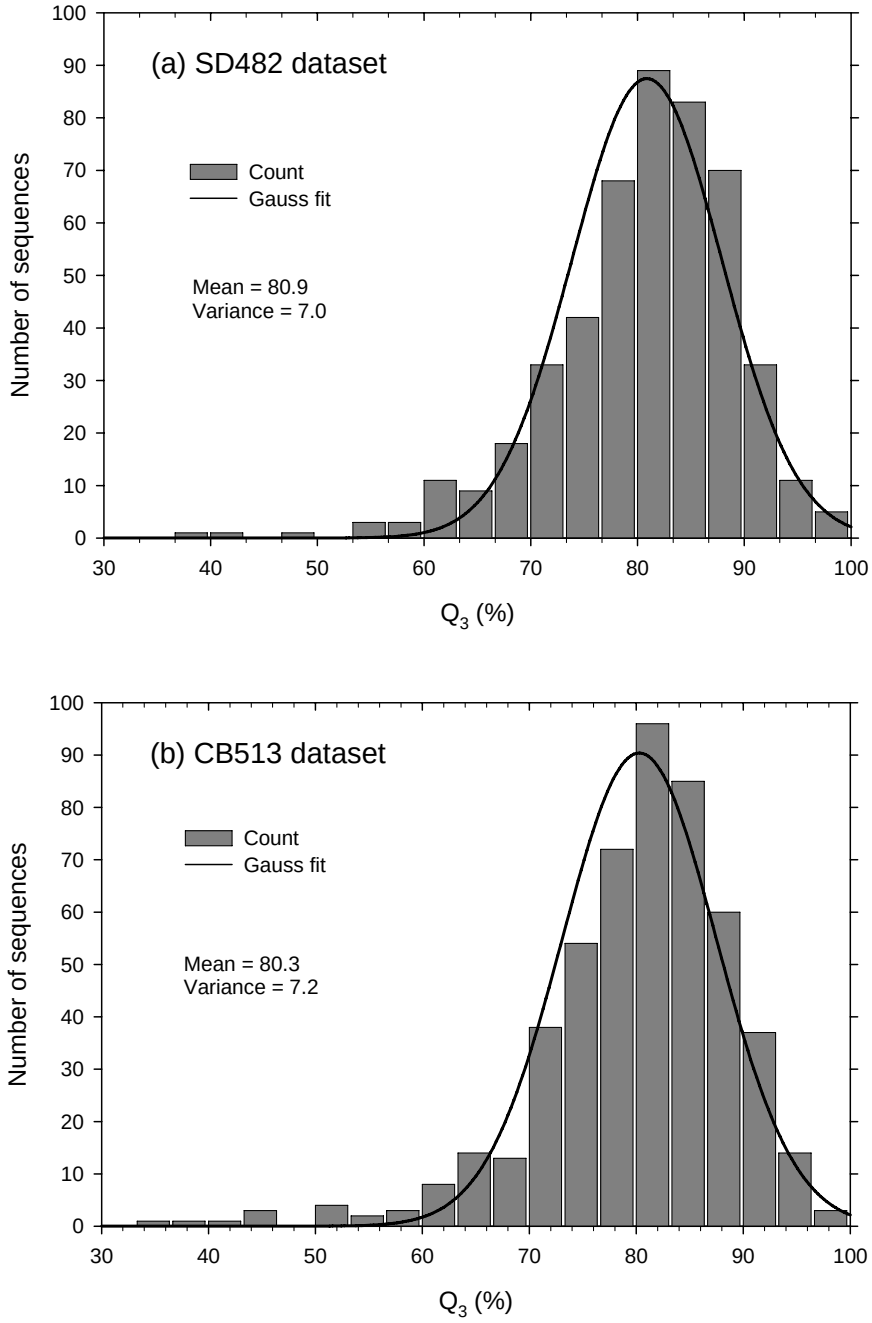


Figure 4-5. Histogram of three-state accuracy (Q_3) scores per protein sequence for TreeCRFpsi: (a) on SD482 dataset; (b) on CB513 dataset. Results were based on 7-fold traditional cross validation under the same conditions as Tables 4-3 and 4-4, respectively.

Table 4-5. Number of sequences correctly predicted at different thresholds. Results were based on 7-fold traditional cross validation under the same conditions as Tables 4-3 and 4-4, respectively.

Datasets	# of sequences correctly predicted better than 76.7%	# of sequences correctly predicted between 70.0% and 93.3%
SD482	74.7%	89.0%
CB513	72.3%	88.9%

4.6 Comparisons of Results from TreeCRFpsi and Other Prediction Models on the CB513 Dataset

Table 4-6 shows the results reported on the CB513 dataset for different state-of-the-art prediction methods. The CRF models outperform other models. Our TreeCRFpsi model exhibited the best three-state prediction score (0.6% better than the second best), and especially helix (2.3% better) and strand (1% better) scores. Although the SOV99 score of our TreeCRFpsi model was a little worse than Q_3 score, it's comparable with others.

Further analysis via the unpaired differences test (Dietterich, 1998) for two error rates (e.g., Q_3 for TreeCRFpsi (0.776) vs. marginalCRFpsi (0.770)) showed that the difference is statistically significant at the 0.001 level. Specifically, a 99.90% confidence interval for the difference between these two proportions does not contain zero, so treeCRFpsi is statistically significantly better:

99.90% confidence interval: $0.001303 <= p_1-p_2 <= 0.010697$

However, we cannot apply a stronger test without having the residue-by-residue predictions of each method.

Table 4-6. Comparisons of prediction results from different methods on the CB513 dataset. Results were based on 7-fold traditional cross validation.

Method	**Q_3 (%)**	**Q_H (%)**	**Q_E (%)**	**Q_C (%)**	**SOV99 (%)**
SVMfreq[a]	73.5	75.0	60.0	79.0	(76.2)[f]
SVMpsi[b]	76.6	78.1	65.6	81.8	73.5
PSIMLR[c]	76.4	79.1	64.7	80.5	73.2
MEMMs[d]	76.9	78.3	62.2	83.3	(76.1)[f]
marginalCRFpsi[d]	77.0	78.3	63.4	83.4	(76.2)[f]
TreeCRFpsi[e]	**77.6**	**80.6**	**66.1**	81.2	**73.9**

[a] Hua and Sun (2001); [b] Kim and Park (2003); [c] Qin *et al* (2005); [d] Liu *et al* (2004); [e] based on CB513 dataset (this work); [f] the authors didn't state SOV94 or SOV99.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Protein secondary structure prediction has been studied for almost half century. About one decade ago, with the incorporation of position-specific scoring matrix profiles generated from PSI-BLAST against some large protein sequence database combined with some advanced algorithms, the prediction accuracy broke though 70%. Position-specific scoring matrix profiles significantly improved the prediction accuracy compared to results without PSSMs on the same TreeCRF method (Ashenfelter, 2003; Dietterich *et al.*, 2004).

TreeCRFs are competitive and possibly slightly more accurate than other existing methods. CRFs provide an elegant probabilistic model, which no other state-of-the-art method can provide. This is particularly useful if the secondary structure predictions are to be used as input to subsequent processing, because the predicted probabilities have well-understood semantics, unlike neural net or SVM output scores.

5.2 Future Work

Although our overall three-state Q_3 prediction accuracy is the best among the published results, the segment overlap (SOV) measure is not as satisfactory as Q_3. Information theory analysis shows that the correlation between neighboring secondary structures are much stronger than that of neighboring amino acid residues (Crooks and Brenner, 2004). Therefore, SOV scores have more structural meaning. In this work, we applied the forward-backward algorithm to train the TreeCRF. Using Viterbi algorithm instead might

increase the SOV measure, since Viterbi selects the best combined prediction for the whole protein sequence.

Now that we have a robust and efficient TreeCRFpsi method, a web server to provide protein secondary structure prediction services would be a good idea in the near future. Some work has to be carried out to make it automatic, by integrating all the steps from accepting protein sequences, generating PSSM raw profiles, conducting a series of steps of transformation, feeding to TreeCRF, to analyzing and reporting the predicted results.

Although the accuracy is close to 80% for current *ab initio* prediction from amino acid sequences, it is still not very useful in practice for 3D structure prediction, since the single sequence prediction accuracy can not be guaranteed to be high. Apparently, the evolutionary information generated from PSI-BLAST is not enough. To improve the prediction accuracy, some other information should be incorporated, such as dihedral angle restrictions (Wood and Hirst, 2005), solvent accessibility (Kim and Park, 2004; Qin *et al.*, 2005), NMR chemical shifts (Hung and Samudrala, 2003), disulfide bonding patterns (Taskar *et al.*, 2005), and so on.

As pointed out in Chapter 1, protein secondary structure prediction is just an intermediate step toward predicting protein structure and function. Our ultimate goal is to predict the three-dimension folding (the tertiary structure and/or quaternary structure) (Zhang, 2002). However, this is much more difficult because higher-order folding depends so much on specific side chain interactions, often between residues far away from one another in the sequence.

BIBLIOGRAPHY

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389-3402.

Ashenfelter, A.J. (2003). Sequential supervised learning and conditional random fields. M.S. Thesis, Dept. Computer Science, Oregon State University, Corvallis, OR.

Bystroff, C. and Shao, Y. (2002). Fully automated ab initio protein structure prediction using I-SITES, HMMSTR and ROSETTA. *Bioinformatics*. **18** Suppl 1, S54-S61.

Chou, P.Y., and Fasman, G. (1974). Conformational parameters for amino acids in helical, sheet, and random coil regions calculated from proteins. *Biochemistry*. **13**, 211-222.

Crooks, G.E. and Brenner, S.E. (2004). Protein secondary structure: entropy, correlations and prediction. *Bioinformatics*. **20**, 1603-1611.

Cuff, J.A. and Barton, G.J. (1999). Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins.* **34**, 508-519.

Cuff, J.A. and Barton, G.J. (2000). Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins.* **40**, 502-511.

Dietterich, T.G., (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, **10**, 1895-1924.

Dietterich, T.G. (2002). Machine learning for sequential data: A review. *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 15-30, Springer Verlag, NY.

Dietterich, T.G., Ashenfelter, A., and Bulatov, Y. (2004). Training conditional random fields via gradient tree boosting. *Proceedings of the 21st International Conference on Machine Learning*, 2004, pp. 217-224, Banff, Canada.

Eddy, S.R. (2004). Where did the BLOSUM62 alignment score matrix come from? *Nat. Biotechnol.* **22**, 1035-1036.

Finer-Moore, J.S., Maley, G.F., Maley, F., Montfort W.R., and Stroud R.M. (1994) Crystal structure of thymidylate synthase from T4 phage: component of a deoxynucleoside

triphosphate-synthesizing complex. *Biochem.*, **33**, 15459-68.

Guo, J., Chen, H., Sun, Z., and Lin, Y. (2004). A novel method for protein secondary structure prediction using dual-layer SVM and profiles. *Proteins.* **54**, 738-743.

Henikoff, S. and Henikoff, J.G. (1992). Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U. S. A.* **89**, 10915-10919.

Hua, S. and Sun, Z. (2001). A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach. *J. Mol. Biol.* **308**, 397-407.

Hung, L.H. and Samudrala, R. (2003). PROTINFO: Secondary and tertiary protein structure prediction. *Nucleic Acids Res.* **31**, 3296-3299.

Jones, D.T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.* **292**, 195-202.

Jones, D.T. and Swindells, M.B. (2002). Getting the most from PSI-BLAST. *Trends Biochem Sci.* **27**, 161-4

Kabsch, W., and Sander, C. (1983). Dictionary of protein secondary structure: Pattern recognition of hydrogen bonded and geometrical features. *Biopolymers* **22**, 2577-2637.

Karplus, K., Barrett, C., Cline, M., Diekhans, M., Grate, L., and Hughey, R. (1999). Predicting protein structure using only sequence information. *Proteins.* **S3**, 121-125.

Kendrew, J. C., Dickerson, R. E., Strandberg, B. E., Hart, R. J., Davies, D. R., and Phillips, D. C. (1960). Structure of myoglobin: A three-dimensional Fourier synthesis at 2 Å resolution. *Nature.* **185**, 422-427.

Kim, H. and Park, H. (2003). Protein secondary structure prediction based on an improved support vector machines approach. *Protein Eng.* **16**, 553-560.

Kim, H. and Park, H. (2004). Prediction of protein relative solvent accessibility with support vector machines and long-rang interaction 3D local descriptor. *Proteins.* **54**,557-562.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th International Conference on*

Machine Learning (pp. 282-289). Morgan Kaufmann, San Francisco, CA.

Lesk, A. M. (1991). *Protein architecture - A practical approach*. Oxford University Press, Oxford, New York, Tokyo.

Liu, Y., Carbonell, J., Klein-Seetharaman, J., and Gopalakrishnan, V. (2004). Comparison of probabilistic combination methods for protein secondary structure prediction. *Bioinformatics*. **20**, 3099-3107.

Mathews, C.K., van Holde, K.E., and Ahern, K. (2000). *Biochemistry*, Third Ed. Benjamin Cummings, San Francisco, CA

McCallum, A., Freitag, D., & Pereira, F. (2000). Maximum entropy Markov models for information extraction and segmentation. *Proc. 17th International Conf. on Machine Learning*, pp. 591-598, Morgan Kaufmann, San Francisco, CA.

Mitchell, E. M., Artymiuk, P. J., Rice, D. W. & Willett, P. (1992). Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.* **212**, 151-166.

Nelson, D.L. and Cox, M.M. (2004). *Lehninger principles of biochemistry*, Fourth Ed. W. H. Freeman, New York, NY.

Pauling, L. and Corey, R.B. (1951). Configurations of polypeptide chains with favored orientations around single bonds: Two new pleated sheets. *Proc. Natl. Acad. Sci. U.S.A.* **37**, 729-740.

Pauling, L., Corey, R.B., and Branson, H.R. (1951). The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain. *Proc. Natl. Acad. Sci. U.S.A.* **37**, 205-234.

Perutz, M. F., Rossmann, M. G., Cullis, A. F., Muirhead, G., Will,G., and North, A.T. (1960). Structure of haemoglobin: A three-dimensional Fourier synthesis at 5.5 A resolution, obtained by X-ray analysis. *Nature.* **185**, 416-422.

Petersen, T.N., Lundegaard, C., Nielsen, M., Bohr, H., Bohr, J., Brunak, S., Gippert, G.P., and Lund, O. (2000). Prediction of protein secondary structure at 80% accuracy. *Proteins. IIS-0307592* **41**, 17-20.

Przybylski, D. and Rost, B. (2002). Alignments grow, secondary structure prediction improves. *Proteins.* **46**, 197-205.

Qian, N. and Sejnowski, T.J. (1988). Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.* **202**, 865-884.

Qin, S., He, Y., and Pan, X.M. (2005). Prediction protein secondary structure and solvent accessibility with an improved multiple linear regression method. *Proteins.* **61**, 473-480.

Rost, B. (1996). PHD: predicting one-dimensional protein structure by profile based neural networks. *Meth. Enzymol.* **266**, 525-539.

Rost, B. (1998). Twilight zone of protein sequence alignments. *Protein Eng.* **12**, 85-94

Rost, B. and Sander, C. (1993). Secondary structure prediction of all-helical proteins in two states. *Protein Eng.* **6**, 831-836.

Rost, B. and Sander, C. (2000). Third generation prediction of secondary structure, *in* Webster D. (Ed) Protein Structure Prediction: Methods and Protocols, pp. 71-95, Humana Press, Clifton, NJ.

Rost, B., Sander, C., and Schneider, R. (1994). Redefining the goals of protein secondary structure prediction. *J. Mol. Biol.* **235**, 13-26.

Russell, S. and Norvig, P. (2003). *Artificial intelligence: A modern approach*, 2nd Ed. Prentice Hall, Upper Saddle River, NJ.

Schäffer, A.A., Aravind, L., Madden, T.L., Shavirin, S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., and Altschul, S.F. (2001). Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucleic Acid Res.* **29**, 2994-3005.

Shen, R. and Dietterich, T.G. (2006). A new methodology on cross validation. *J. Machine Learning Res*. In preparation.

Szent-Györgyi, A. G., and Cohen, C. (1957). Role of proline in polypeptide chain configuration of proteins. *Science.* **126**, 697.

Taskar, B., Chatalbashev, V., Koller, D., and Guestrin, C. (2005). Learning structured prediction models: A large margin approach. *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany.

Wallach, H.M. (2003). Efficient training of conditional random fields. *Proceedings of the*

6th Annual CLUK Research Colloquium, Edinburgh, U.K.

Ward, J.J., McGuffin, L.J., Buxton, B.F., and Jones, D.T. (2003). Secondary structure prediction with support vector machines. *Bioinformatics*. **19**, 1650-1655.

Wood, M.J. and Hirst, J.D. (2005). Protein secondary structure prediction with dihedral angles. *Proteins.* **59**, 476-481.

Yi, T.M. and Lander, E.S. (1993). Protein secondary structure prediction using nearest-neighbor methods. *J. Mol. Biol.* 232, 1117-1129.

Zemla, A., Venclovas, C., Fidelis, K., and Rost, B. (1999). A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment. *Proteins.* **34**, 220-223.

Zhang, H. (2002). Protein tertiary structures: Prediction from amino acid sequences. *Encyclopedia of life sciences*, pp. 1-7, Macmillan Publishers Ltd, Nature Publishing Group.