Application of Maximum Entropy Markov Models on the Protein Secondary Structure Predictions

Yohan Kim
Department of Chemistry and Biochemistry
University of California, San Diego
La Jolla, CA 92093
ykim@ucsd.edu

Abstract

An application of a new probabilistic modeling framework, Maximum Entropy Markov Model, on the protein secondary structure prediction problem is described [6]. As in previous domains of problem, such as the task of segmenting a body of text, within which MEMM was applied [6], the secondary structure prediction problem requires labeling an observation sequence of alphabets. This paper is an exploratory effort that attempts to establish the feasibility of using the new framework on this long-standing problem. Our initial results produced with a rather simple MEMM model show promise (≈ 58% accuracy) and suggest directions for further improvements.

1 Introduction

The problem of predicting the secondary structure of proteins has been around even before the structure of first protein was solved by the x-ray crystallography method [1]. The database collecting those structures solved since is a testament to the fact that there exist recurring shapes representing various parts of the protein whose geometries are guided by the composition of the amino acid sequence [7]. Initially these recurring shapes were given secondary structure labels by the human experts in the area. However this method of labeling introduced subjectivity. In 1983, Kabsch *et al* introduced the DSSP program that consistently assigned secondary structure labels to the solved structures. The program bases its method of labeling on the hydrogen bonding patterns found in the solved structure.

Since the growth of database holding only protein sequences outpaced that of solved protein structures, much effort has been made in the area of predicting structures from the amino sequences. One form of predicting the protein structure from the amino acid sequence is the secondary structure prediction. Instead of predicting the full 3-D coordinates of the structure, the task is to predict a sequence of secondary structure labels based on the amino acid sequence alone. Present work uses the set of secondary structure labels whose size is 3 (i.e. Helix, Coil, and Sheet).

Previous works on predicting secondary structures of proteins have yielded the best percent accuracy ranging from 63% to 71% [8]. These numbers, however, should be taken with caution since performance of a method based on a training set may vary when trained on a different training set.

MEMM was recently introduced to model sequential data [6]. Some of the examples that this new framework can be applied include tagging parts-of-speech on to a body of a text, labeling segments of questions and answer on Frequently-Asked-Questions list (FAQs), and others. These types of problem are so called 'Input-Output' problem. The essential task is to take an input sequence and produce an output sequence that contains labels of the corresponding parts of the input sequence. The protein secondary structure prediction is another 'Input-Output' problem. In this case, the task is to label a given sequence of amino acids, X, with a sequence of secondary structure labels, S. Here, the alphabet sizes of X and S are 20 and 3, respectively.

MEMM shows better performance than the Hidden Markov Models (HMM) at least in the problem domains that it was initially implemented. Although the work that followed [5] showed that MEMM suffers from the 'label-bias' problem and it introduced yet another model, implementing MEMM was driven by following motivations:

1. offers a new testing ground to tackle the well established problem (the protein secondary structure prediction) and shed light on MEMM's performance
2. allows overlapping features (features will be precisely defined later in the paper)
3. it is a conditional model

No previous application of MEMM on the secondary structure prediction problem has been found in the literature. With the potential advantage that MEMM offers overlapping features and that it is a conditional model dependent only on the observation sequence, present work introduces the new modeling framework to a long-standing problem of predicting the protein secondary structure from an amino acid sequence.

The paper focuses on how MEMM was used in the secondary structure prediction problem. Present implementation of MEMM differs in two aspects from the previous ones. Notably there are no state transition probability functions but a single function that assigns probability value to any possible combination of state and fragment. This implies that Viterbi algorithm is not required to infer the state sequence. In addition, types of feature functions that were chosen are different from the ones chosen for, say for instance, segmenting FAQ problems. This is due to the basic difference in the nature of the problems. In the later part of the paper, a general overview of how the model is trained is given and the results of the application of MEMM on Salzberg's data set are shown [9].

2 Maximum Entropy Markov Model

A summary of the new modeling framework introduced is given [6]. The essential principle behind Maximum Entropy approach is to create a model that satisfies all known constraints but otherwise treat the unknowns uniformly [2]. The final model after training is a collection of separately trained *transition* functions, $P_{s'}(s|x)$ and a vector of parameters λ. The trained function $P_{s'}(s|x)$ outputs a probability value of seeing a state transition from s' to s given observation x. When $P_{s'}(s|x)$ is summed over s, the sum should be 1 by the definition of the probability function.

$$P_{s'}(s \mid x) = \frac{1}{Z(s', x)} \exp\left(\sum_a \lambda_a f_a(s, x) \right) \quad (1)$$

Z(s',x) in equation (1) is a normalizing factor. Features $f_a(s,x)$ are binary functions that capture the important relationship among state and observation sequences.

Present work does not explicitly incorporate the state transitions. In other words, no functions of $P_{s'}(s|x)$ are constructed. Instead, only one probability function $P(s|x)$ is used, and the observation *x* is characterized by a set of 300 feature functions. The decision to use only one probability function was forced by the choice and the total number of feature functions. However, it should be noted that the feature functions employed in this work implicitly capture the dependence on previous states. Detailed discussions on feature functions will be given later.

The absence of state transition functions makes the use of Viterbi algorithm to infer the state sequence unnecessary. One can infer the state sequence just by picking the state label at each positions of the sequence that has the highest probability based on the $P(s|x)$.

3 **Feature Selection**

One last important aspect of the current implementation is the task of selecting the set of features that capture the important relationships among state and observation sequences. No systematic effort was made to select the best set of feature functions. Selecting a set of features was made difficult by the uncertainty of how many features and what kinds are needed to sufficiently capture the relationship among the amino acid sequence and the corresponding secondary structure labels. In addition, due to long training time needed, only a small number of combinations of feature functions were explored.

Another difficulty in choosing features was due to inherent difficulty in understanding what the language of amino acids is. To give an analogy using text-segmenting FAQs as an example, some features such as *indent-present* and *contains-question-mark* are immediately understood by the reader to have important roles in segmenting FAQs into question and answer pairs. The reader has an intuitive notion that the two features mentioned have some relevance in identifying different segments of FAQs. This intuition is made possible since the reader understands what each of the observation sequence means (in this case, a body of English text). To extend the analogy of the reader and the observation sequence, the reader in the case of secondary structure prediction is not human but a dense collection of cellular machineries that understand what the amino acid sequence means and that carry out myriads of operations dictated by the amino acid compositions.

In the end, the set of features that are described by [3] are used. This set of features is simple, and one can immediately see relationships among state and observation sequence that it is trying to capture. The present work uses 300 feature functions that are succinctly represented by the following equation:

$$f_k(s_i, x_i) = \begin{cases} 1 \text{ if } table(k) \text{ matches } s_i \text{ and } x_i \\ 0 \text{ if otherwise} \end{cases} \quad (2)$$

Table 1: First 6 entries of *table*(*k*).

k	*s*	*aa*	*p*
1	1	1	1
2	1	1	2
3	1	1	3
4	1	1	4
5	1	1	5
6	1	2	1

Table1: State labels and amino acids are represented with integers.
k = index of the feature function
i = position in the observation sequence
s_i = possible state assignment at position *i*. $|s| = 3$.
x_i = possible amino acid sequence fragment of length 5 at position *i*.
aa = amino acid index ranging from 1 to 20.
p = position of the amino acid within an amino acid sequence fragment.

Two examples using this set of feature functions $f_1(1, [1\ 1\ 1\ 1\ 1]) = 1$ and $f_1(1, [2\ 1\ 1\ 1\ 1]) = 0$ are provided to clarify the way in which these feature functions are used. The feature function with an index $k = 1$ means that this function returns a value of 1 only if the arguments (i.e. s_i and an amino acid at position *p* of the fragment x_i) of the function match label and amino acid stored in the *table* at *k*-th row (Figure 1). In the first example, $s_i = 1$ and $x_i = [1\ 1\ 1\ 1\ 1]$. Having $k = 1$ means that this feature function returns 1 only when it sees a state that has a value of 1 and an amino acid at position 1 within fragment x_i with value of 1. In the second example, the feature function sees a state value of 1; however, it sees an amino acid value of 2 at position 1 of fragment $x_i = [2\ 1\ 1\ 1\ 1]$. Consequently, the latter feature function yields 0.

This set of feature functions attempts to capture, borrowing the term used in [3], the amount of 'information' that an amino acid at position *p* has on the state assignment of the amino acid at the middle of the window (Figure 2). There are 3 possible secondary structure labels, 20 possible amino acids, and 5 possible positions. This yields the total number of 300 unique permutations possible for the triplet (label, amino acid, position) where each permutation is captured by one feature function.

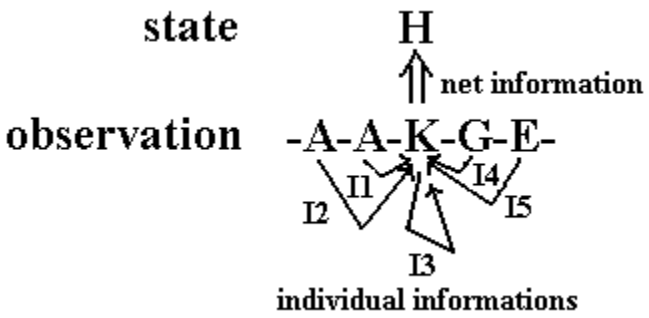

Figure 2: Individual amino acids at various positions of a fragment contributing 'information' to yield the final state assignment for the middle amino acid. An amino acid sequence fragment of length 5 is presented with its middle residue labeled as H or Helix. Each one of I1, I2, I3, I4, and I5 is represented by a feature function. Since there are 3 states, 20 amino acids, and 5 positions, the total number of feature functions is 300.

4 Parameter Estimation with Generalized Iterative Scaling

The MEMM model is trained by maximizing the log probability of the state sequence given the observation sequence. This implies that the empirical expectations of the features have to equal to those calculated using the trained model. This condition is shown by the equation (3).

$$\frac{1}{N}\sum_{i=1}^{N} f_k(s_i, x_i) = \frac{1}{N}\sum_{i=1}^{N}\sum_{s_i \in S} P(s_i \mid x_i) f_k(s_i, x_i) \qquad (3)$$

N = total length of the training sequence of state and observation pairs.

The term on the left is the empirical expectation of the feature function $f_k(\ )$. The term on the right is the expectation calculated using the model.

This work follows the same outline of the algorithm provided in earlier work to estimate the parameters [6]. One minor difference to note is that the arbitrary constant C was set to 1. The trained MEMM model is a set of converged parameters λ and the single probability function $P(s|x)$. It should be noted that the final values of λ indicate how much each feature function influence in determining the probability of s given x. Roughly speaking, greater the value of a parameter, the more it allows the corresponding feature function to contribute to the final probability value.

5 Experimental Results

A data set used in earlier works [8] provided both the training and testing data sets. Briefly, this data set contained amino acid sequences of experimentally solved structures and corresponding sequences of secondary structure labels as determined by the DSSP program [4].

The original data set was processed to fit the needs of the model. Pairs of amino acid sequences and secondary structure label sequences within the original data set

were combined into one file. Then, another file was generated that had two columns:

1. first column was allotted for the fragments of amino acid sequence derived by sliding a window of length 5 along the amino acid sequence
2. second column was for the state labels assigned to the middle residues of the corresponding fragments.

The processed file contained total of 18,790 lines, each containing a fragment-label pair. The training set consisted of roughly half of the processed file. The test set was the latter half. MEMM was implemented within Matlab environment. Training was stopped after 4 iterations (parameters converged to within second decimals), and the entire training took approximately 16 hours on Pentium II 400 Mhz.

Table 2**:** Comparisons of % correct labels among various methods.

Methods	% correct
PEBLS (1992)	71
Zhang et al (1992)	66.4
Qian & Sejnowski (1988)	64.3
Holley & Karplus (1989)	63.2
MEMM	**58.55**
random	39.9

Table 2 is shown to illustrate the range in which previously developed methods performed relative to MEMM. These numbers are not directly comparable since different training sets were used. In addition relatively poor performance of MEMM may be attributed to the fact that it only took into account 2 neighboring amino acids whereas other methods such as PEBLS and Qian & Sejnowski used 8 neighboring amino acids or more. Salzberg *et al* reported that their method, PEBLS, considered 9 neighboring amino acids because they found this number to be optimal.

```
HHHHHHHCHHHHHHHHHCCCCHHHHHHHHHHHHCCCCCEEEEECCCEEECCEEECCCCEEEECHHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CHHHHHCCCCCHHHHHCCCCCCCCCHHHHHHEHEEECCCCEECCCCHECCHEHCCCCHHECCHCECCCCHCHCCCCCCECCCCCCCCCCCCCCCCHHHC
```

Figure 3**:** One-to-one pairing of the correct sequence label at top and MEMM predicted sequence at the bottom for the first 100 fragments in the test set.

MEMM predicts a rough outline of the patches of continuous secondary structure labels. In this particular example, for two long stretches of continuous secondary structures (first 1/3 and last 1/3 of the sequences shown above), MEMM gets majority of the labels correct. In the middle part of the sequences, where correct labels fluctuate more often than the other regions, MEMM performs poorly.

6 Conclusions and Future Directions

A new probabilistic modeling framework was applied to the protein secondary structure prediction problem. The set of simple features introduced by [3] was used. The results of the experiment on Salzberg's data set show some promises (≈ 58% correct labels). This is over 20% better than randomly assigning the labels with empirical distribution of secondary structure labels in the data set (H = 26.1 %; C = 54.4%; E = 19.5 %). The results of the experiment should be viewed with the knowledge that only fragments of length 5 were considered in the model. As was noted in [8], a better fragment length than the one used here can probably be found for the current choice of feature set. More over, many variations on the selection of features are possible. Present work did not take the advantage of overlapping features offered by MEMM.

Although MEMM offers interesting aspect to modeling the sequential data, the experience with implementing the model suggests that the architecture of MEMM puts much constraint on the number of features that can be used during training. Greater the number of features, the calculation of expectation value of each feature function increases correspondingly (see the outline of algorithm in [6]). With training set of size N and feature set of size M, the time complexity of the training is $O(j*M*N)$, where j is the number of iterations. The number of feature functions can be significantly decreased by introducing overlapping feature functions. Much room for improvements exists in the area of feature selection.

7 References

[1] Baldi, P. Bioinformatics. (2001). The Machine Learning Approach. 2nd ed.

[2] Berger, A. L., Della Pietra, S. A., & Della Pietra, V. J. (1996). A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*. 22, 1.

[3] Garnier, J., Osguthorpe, D. J., & Robson, B. (1978). Analysis of the Accuracy and Implications of Simple Methods for Predicting the Secondary Structure of Globular Proteins. *Journal of Molecular Biology*, 120, 97-120.

[4] Kabsch, W. & Sander, C. (1983). Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features. *Biopolymers*, 22, 2577-2637.

[5] Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proc*. ICML.

[6] McCallum, A., Freitag, D. & Pereira, F. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. *Proc*. ICML.

[7] Protein Data Bank. http://www.rcsb.org/pdb/

[8] Salzberg, S. & Cost, S. (1992). Predicting Protein Secondary Structure with a Nearest-neighbor Algorithm. *Journal of Molecular Biology*. 227, 371-374.

[9] Salzberg's Data Set. http://www.cs.jhu.edu/~salzberg/