Improving protein secondary structure prediction using a simple $k$-mer model

Martin Madera$^1$, Ryan Calmus$^1$, Grant Thiltgen$^2$, Kevin Karplus$^2$ and Julian Gough$^1$*

$^1$Department of Computer Science, Woodland Road, University of Bristol, BS8 1UB, UK
$^2$Department of Biomolecular Engineering, University of California, Santa Cruz, CA 95064, USA.

Associate Editor: Prof. Burkhard Rost

**ABSTRACT**

**Motivation:**
Existing methods for protein sequence analysis are generally first-order and inherently assume that each position is independent. We develop a general framework for introducing longer-range interactions. We then demonstrate the power of our approach by applying it to secondary structure prediction; under the independence assumption sequences produced by existing methods can produce features that are not protein-like, an extreme example being a helix of length one. Our goal was to make the predictions from state of the art methods more realistic, without loss of performance by other measures.

**Results:**
Our framework for longer-range interactions is described as a $k$-mer order model. We succeeded in applying our model to the specific problem of secondary structure prediction, to be used as additional layer on top of existing methods. We achieved our goal of making the predictions more realistic and protein-like, and remarkably this also improved the overall performance. We improve the SOV score by 1.8%, but more importantly we radically improve the probability of the real sequence given a prediction from an average of 0.271 per residue to 0.385. Crucially this improvement is obtained using no additional information.

**Availability:** http://supfam.cs.bris.ac.uk/kmer
**Contact:** gough@cs.bris.ac.uk

1 INTRODUCTION

The prediction of secondary structure remains very important in the field of protein biology, even if the methods have matured and development of the algorithms is a far less active area than a decade ago. One of the reasons for this decline in activity is that most of the competing methods have converged on a similar level of performance beyond which they have been unable to improve, and possibly because the level of performance that they achieve is, by bioinformatics standards, exceptionally good. This is reflected in the fact that the Critical Assessment of Techniques for Protein Structure Prediction (CASP) (Moult \textit{et al}., 1995) competition for protein structure prediction ceased to assess this as an official category some years ago, as has the EVA (Koh \textit{et al}., 2003) continuous benchmarking project. Accurate prediction of secondary structure elements from an amino acid sequence remains very useful to biologists in its own right, but it is worth pointing out that it is also an essential component of tertiary structure prediction, which in contrast is far from solved and continues to be a highly active area of research. In addition, sequence comparison methods have more recently incorporated local structure tracks (such as secondary structure or burial). The extra information utilised by the new methods has led to considerable improvements in fold recognition and alignment accuracy.

There are many different methods for secondary structure prediction, e.g. (Rost, 1996; Jones, 1999; Cuff \textit{et al}., 1998; Ouali and King, 2000; Pollastri and McLysaght, 2005; Katzman \textit{et al}., 2008), all using neural networks. Analysis of the results of the last CASP competition to include secondary structure (Aloy \textit{et al}., 2003) gives a good indication of the state of the art, and the range of methods available; for this work we chose to use PREDICT-2ND (Katzman \textit{et al}., 2008) which is named as one of the 3 leading original methods. Despite secondary structure prediction methods being able to correctly assign either helix, strand or loop to roughly 80% of the individual positions in a protein sequence, the overall prediction is not protein-like.

For example these methods are capable of predicting a helix of a single amino acid in length (although consequently most have implemented an ad hoc filter to remove them). What we aim to achieve in this paper is to create a model which makes the overall predictions of existing methods more realistic and protein-like without loss of performance as measured on a per-residue basis. It is possible that in doing this we may improve prediction accuracy, even if it is not our original goal.

All current protein sequence comparison methods, when making a prediction, implicitly assume in one way or another that each position in a protein sequence is independent of other positions in the sequence. In the case where a sliding window is used to capture information from the neighbourhood of an individual position, the decision of what to predict at that position is still made independently of other predictions in the neighbourhood. While the independence assumption is an acceptable approximation when comparing individual amino-acid sequences, it fails dramatically for many local structure alphabets. To give a specific example, when a position in a sequence is an $\alpha$-helix, the adjacent positions

---

*to whom correspondence should be addressed
are highly likely (\(\sim 90\%\)) chance to also be \(\alpha\)-helical. In fact we have observed three broad types of correlations that violate the independence assumption:

1. 

**Short-range.** If it stands for a helix and \(A\) for an anti-parallel strand, there are three times more occurrences of \(\alpha H\), 14x more occurrences of \(HA\), and at least 10,000x fewer occurrences of \(HA\) and \(AH\) than one would expect based on the frequencies of \(H\) and \(A\) under the independence assumption.

2. 

**Medium-range.** The lengths of helices, strands and loops form well-defined distributions with exponential tails. The points at which exponential decay sets in are different for each structural type, and so are the decay constants. Further, adjacent secondary structure elements are frequently of a comparable physical length; for example, we found that a strand of length 4 is followed by another strand of length 4 twice as often as it is followed by a strand of length 5, even though both lengths occur with roughly the same frequency.

3. 

**Long-range.** If a residue lies within a parallel strand, a strand residue 100 residues away is roughly six times more likely to also be in a parallel strand than if the first position were an anti-parallel strand.

As detailed below, in this paper we concentrate in this first instance on using the first of these three correlations to our advantage, partly because they are amenable to exact analysis to verify the results. We have however ensured that in formulating our approach we developed a model general enough to be applied to the other two, and in principle other higher order sequence information at medium and long range. Please see the discussion of the method for more detail.

More specifically the problem we have chosen as a starting point is that of sampling from a profile of secondary structure sequences, e.g. one generated by neural networks for structure prediction. We can measure from real sequences how often we observe each individual amino acid (1-mer), each possible pair (2-mer), or every combination of up to \(k\) amino acids (\(k\)-mer). Our goal is to change sequence probabilities to reward \(k\)-mers that are typically under- or over-predicted compared to real sequences, and penalise \(k\)-mers that are over- or under-predicted, so that sequences sampled from the modified system look protein-like across \(k\) amino acids.

Here we present a Conditional Random Field (CRF) (Lafferty et al., 2001) model as a solution to the problem. CRFs have previously been used in bioinformatics (Sato and Sakakibara, 2005; Do et al., 2006; Liu et al., 2004) and may be gaining popularity. Protein amino-acid sequences have traditionally been handled with hidden Markov models (HMMs) (Madera and Gough, 2002), but except for HMMSTR (Bystroff et al., 2000) and a more recent attempt by Krogh (Won et al., 2007), they have not made much of an impact in secondary structure prediction. This is because traditional 1st-order HMMs cannot handle very well the sorts of overlapping long-range features that are necessary for a good model of local structure. CRFs are an appropriate response to precisely this shortcoming of HMMs.

## 2 METHODS

### 2.1 A \(k\)-mer model of correlated sequences

As a preparation for our full model, we start with a formulation of a simple Markov chain of order \(n - 1\) in terms of log-odds scores. This formulation will play a key role in the full model.

Let a sequence \(y\) of length \(L\) be denoted \(y_1\ldots L\), and a sub-sequence of \(y\) be denoted \(y_m\ldots n\). Let us now suppose that sequences in some large training dataset \(T\) can be modelled as Markov chains of order \(n - 1\), i.e. that

\[
P(y) = P(y_{n-1}) P(y_n|y_{n-1}) P(y_{n+1}|y_{n-1}y_n) \ldots P(y_L|y_{L-n+1}\ldots L-1).
\]

(1)

We can express the individual probabilities in (1) in terms of the distribution of \(k\)-mers in \(T\). Let us denote the relative frequency of a \(k\)-mer \(a\) in \(T\) by \(T_k(a)\), where for each value of \(k\) the relative frequencies of all \(k\)-mers sum to one. The probability of the initial \((n - 1)\)-mer is then simply

\[
P(y_{n-1}) = T_{n-1}(y_{n-1}).
\]

and the transition probabilities are

\[
P(y_m|y_{m-n+1}\ldots m-1) = \frac{T_n(y_{m-n+1}\ldots m)}{T_{n-1}(y_{m-n+1}\ldots m-1)}.
\]

(3)

Substituting (2) and (3) into (1) gives \(P(y)\) in terms of the distribution of \(k\)-mers \(T\), so we shall henceforth denote it by \(P(y|T)\).

We have noticed that \(P(y|T)\) can also be expressed in the following alternative form:

\[
\ln P(y|T) = \mathbf{S} \cdot \mathbf{F}(y),
\]

(4)

where \(\mathbf{P}\) is a feature vector of \(k\)-mer counts and \(\mathbf{S}\) are the corresponding \(k\)-mer scores, defined as follows:

\[
S_1(a_1|T) = \ln T_1(a_1)
\]

(5)

\[
S_2(a_1, a_2|T) = \ln \frac{T_2(a_1, a_2)}{T_1(a_1)T_1(a_2)}
\]

(6)

\[
S_3(a_1, a_2, a_3|T) = \ln \frac{T_3(a_1, a_2, a_3)}{T_2(a_1, a_2)T_2(a_2, a_3)}
\]

(7)

and so on up to \(n\)-mers. An important aspect of this formulation is that Eq. (6,7) can be understood as log-odds scores, in the following sense. For \(k > 1\) the denominator is in fact the frequency of the \(k\)-mer under a \((k - 1)\)-mer model, so

\[
S_k(a_1\ldots k|T) = \ln \frac{T_k(a_1\ldots k)}{P(a_1\ldots k|T_{k-1})}.
\]

(8)

In other words, the score is the log-ratio of the observed frequency of a given \(k\)-mer to its expected frequency under a \((k - 1)\)-mer model.

### 2.2 Correcting profile emissions with a \(k\)-mer model

We apply the formulation in the previous subsection to the problem of generating realistic emissions from secondary structure profiles. Our goal here is to down-weight sequences with \(k\)-mers that are frequently produced in profile samples but occur rarely in real sequences, and conversely to up-weight \(k\)-mers that are sampled less frequently than they occur in real sequences.

We define a profile \(X\) as a sequence of \(L\) probability vectors \(X_1\ldots X_L\), where \(P(a|X_l)\) gives the probability of observing the letter \(a\) at position \(l\) of a sequence emitted from the profile. The total probability for a sequence \(y\) to be emitted from the profile is then

\[
P(y|X) = \prod_{l=1}^{L} P(y_l|X_l).
\]

(9)

Our approach is to modify this emission probability by introducing a joint profile + \(k\)-mer model \(\mathbf{M}\),

\[
P(y|M) = \frac{1}{Z(X, R)} \exp \{ \ln P(y|X) + R \cdot \mathbf{F}(y) \},
\]

(10)
\[ Z(X, R) = \sum_{y_i} \exp \left\{ \ln P(y_i | X) + R \cdot F(y_i) \right\} \]  
\[
\text{(11)}
\]

is the normalisation factor (also called the partition function) and \( R \) is a set of \( k \)-mer scores. The challenge is to come up with scores that would make the distribution of \( k \)-mers in sequences sampled from the joint model \( M \) as close as possible to the training distribution \( T \).

We have discovered that the following simple iterative procedure converges on the right answer:
\[
R_k^{(i)}(a) = 0 \quad \text{(12)}
\]
\[
R_k^{(i)}(a) = R_k^{(i-1)}(a) + S_k(a|T) - S_k(a|B^{(i-1)}). \quad \text{(13)}
\]

Here we use the superscript \( i \) to denote variables pertaining to iteration \( i \) of the model, and \( B^{(i-1)} \) is the distribution of \( k \)-mers observed in a large set of sequences \( B^{(i-1)} \) sampled from iteration \( i - 1 \) of the model.

The \( S_k \) scores in (13) are undefined when \( a \) is absent from \( T \) or \( B^{(i-1)} \). We deal with this in one of the following two ways: (i) When \( a \) is absent from both sequence sets, we simply set \( R_k^{(i)}(a) \) to \( R_k^{(i-1)}(a) \). (ii) When \( a \) is absent from one set but not the other (without loss of generality, let us assume that it is absent from \( T \) but present in \( B^{(i-1)} \)), we reset \( R_k^{(i)}(a) \) to satisfy
\[
S_k(a|T) - S_k(a|B^{(i-1)}) = 0. \quad \text{(14)}
\]

If the new value of \( T_k(a) \) is greater than a cut-off value corresponding to an absolute frequency of 0.5 in the sequence set \( T \), we further reset it to the cutoff value. We do not renormalize \( T_k \).

In simple terms, the motivation behind the regularization scheme is to leave \( R_k^{(i)}(a) \) unchanged as much as possible, unless the absence of \( a \) from \( T \) is too stark and demands an adjustment. We have tried traditional approaches such as simple pseudocounts, or pseudocounts based on expectations from \((k-1)\)-mer models, but found that the present algorithm performs considerably better.

We assess convergence of \( B^{(i)} \) towards \( T \) using the Kullback-Leibler relative entropy,
\[
D_k^{(i)} = \sum_{a_1 \ldots k} T_k(a) \log_2 \frac{T_k(a)}{B_k^{(i-1)}(a)}. \quad \text{(15)}
\]

In cases where \( a \) is missing from one or both sequence sets we follow a procedure similar to the one described above: (i) When \( a \) is absent from both sets, the \( T \log Z_{k} \) score is taken to be zero and both \( T_k(a) \) and \( B_k^{(i-1)}(a) \) are kept at zero. (ii) When \( a \) is absent from one set but not the other, we use (14) to reset the zero frequency, subject to the same cut-off as above. Once a decision has been reached on all zero frequencies, both \( T_k \) and \( B_k^{(i-1)} \) are renormalised.

### 2.3 Exact inference for short \( k \)-mer models

For short \( k \)-mers we can perform exact inference in our model (10) using standard dynamic programming algorithms (Durbin et al., 1998). For example, we can use the Viterbi algorithm to calculate the most likely sequence,
\[
y^{\text{V}^	ext{*}} = \arg \max_y P(y|M), \quad \text{\text{(16)}}
\]

or the forward-backward algorithm to perform posterior decoding (also known as marginalisation), which for each position \( i \) computes the letter \( y_i^{\text{post}} \) most likely observed at that position,
\[
y_i^{\text{post}} = \arg \max_{y_i} \sum_{y_1 \ldots y_{i-1} y_{i+1} \ldots} P(y|M). \quad \text{\text{(17)}}
\]

We can also use forward-backward to calculate the partition function \( Z(X, R) \) from (11).

However, these exact algorithms require keeping track of all possible "sticky ends" of length \( n - 1 \). The memory requirements for doing so become prohibitive even for moderately large \( n \), so we need to turn to sampling.

2.4 MCMC sampling from \( k \)-mer models

We used the Metropolis algorithm (Metropolis et al., 1953), which is the oldest and best known MCMC sampling method though not necessarily the most efficient. Using our joint model (10), for each profile we carried out 30 runs of 1000 mutations per position, retaining only the last sequence from each run and discarding all other sequences as burn-in. Note that we do not need to know \( Z \) for sampling, because it cancels out in the Metropolis probability ratio. In retrospect we are aware that the sampling could be done better. An improvement would be made at no additional cost by doing a smaller number of longer sampling runs, and keeping significantly more samples from each run. Also, Gibbs sampling would be more efficient than the Metropolis algorithm.

The computational complexity of the exact posterior calculation will not scale well to long-range models. However, we do not have to calculate the posterior decoding described above to get a good approximation of \( y_i^{\text{post}} \). Instead, we can calculate an estimate \( y_i^1 \) of the posterior decoding by analysing a set of \( d \) samples drawn from the distribution \( P(y|M) \). In this case, \( y_i^1 \) is simply the classification most frequently observed at position \( i \) across the sample set. As \( d \) tends to infinity \( y_i^1 \) becomes equivalent to \( y_i^{\text{post}} \).

2.5 The STR2 alphabet

The UCSC STR alphabet described in Fig. 1 is an enhancement of the DSSP alphabet (Kabsch and Sander, 1983) that was conceived as a response to the observation that parallel and anti-parallel strands exhibit different hydrophobicity patterns. This implies that it should be possible to distinguish between them when predicting secondary structure from sequence. Possibly for this reason, it has been the most successful alphabet at UCSC in protein alignment and fold recognition tests.

\[
\begin{array}{cccccc}
\text{A} & \text{P} & \text{Y} & \text{M} & \text{I} & \text{L} \\
\text{T} & \text{H} & \text{G} & \text{T} & \text{S} & \text{C} \\
\text{Z} & \text{Q} & \text{E} & \text{D} & \text{B} & \text{N} \\
\text{P} & \text{Q} & \text{E} & \text{D} & \text{B} & \text{N} \\
\text{H} & \text{G} & \text{T} & \text{S} & \text{C} & \text{P} \\
\end{array}
\]
2.6 Training and test datasets

The training and test data used to generate and assess the performance of the neural networks providing our \( k \)-mer model with profiles, and the \( k \)-mer model itself, were drawn from a set of 1763 protein chains known as dunbrack-30gc-1763, created by Katzman et al. (Katzman et al., 2008). The set is based upon output from Dunbrack’s PISCES server (Wang and Dunbrack, 2003) containing 1875 chains with a maximum sequence identity of 30% of which 112 were removed; 77 because their chain lengths were less than 50, 26 because the chains were non-globular, and 9 because the chains exhibited very bad clashes as determined by the UNDERTAKER (Karplus, 2009) clash detector.

Katzman et al. used three-fold cross-validation on their dataset to test their neural networks, randomly splitting it into three subsets of 588, 588 and 587 chains and training each one of three networks on two of the subsets whilst testing on the remaining one. We cross-validated correspondingly using the same three training and test subsets to produce three \( k \)-mer models, and the scores we report are averages over these three sets.

2.6.1 Neural network inputs and training protocol

For each chain in the dataset we generated two local structure profiles using the PREDICT-2ND neural networks (Katzman et al., 2008): one using an alignment consisting solely of a guide sequence, describing the amino acid at each position of the target sequence, and one from the alignment generated by SAM-T06 seeded with the guide sequence; we refer to these as single-sequence and alignment inputs to the \( k \)-mer model respectively.

The PREDICT-2ND neural networks feeding predictions to the \( k \)-mer models are 4-layered (others use 2 layers) feed-forward networks taking as input a sliding window of 27 residues’ worth of multiple alignment profile information (that is, for residues \( i - 13 \ldots i + 13 \) centred around the residue for which a secondary structure classification is required; a single output is returned, consisting of 13 \( \text{STR2} \) classification probabilities for the given residue \( i \), one for each letter in the alphabet. The networks and the software with which they may be utilised are available at http://www.soe.ucsc.edu/~karplus/predict-2nd/ and the SAM-T08 (Karplus, 2009) website.

3 RESULTS

Overview

In Fig. 2 we can see an example of the improvement typically obtained by using the \( k \)-mer model to produce secondary structure predictions over sampling directly from the columns of the profile. The rows generated directly from the profile frequently include unrealistic features such as helices or beta strands of only one or two residues in length, and the major secondary structure elements are often fragmented. The \( k \)-mer model can only improve on the profile it is given. For example in Fig. 2 the first strand of the sequence within the profile is evidently incorrect (at the very top) with respect to the true \( \text{STR2} \) sequence (at the very bottom). In this case the \( k \)-mer model has no hope of correcting the prediction because it only has available prior information on secondary structure in general, not specific knowledge of the protein in question.

Table 1 and Table 2 compare the performances (using several different measures) of the secondary structure prediction under various decodings of our \( k \)-mer model for the given profile, versus the original profile-only performance (classification based on maximal probability at each residue position). A profile can be generated by the neural network from a single sequence, or it can take as input a multiple sequence alignment. The results are shown for both single sequence and alignment inputs. The measures presented in the table are computed as the average over all predictions in the test dataset, and are normalised for sequence length.

We observe that whilst achieving the goal of making the predicted secondary structure more realistic, the \( k \)-mer model (sampled decoding) suffers no significant loss of accuracy as assessed by the \( Q_3 \) and \( Q_{13} \) measures; improves somewhat the accuracy according to the SOV measure; and dramatically improves the chance of predicting the real secondary structure sequence. Unsurprisingly the results are all consistently better when using profiles derived from multiple sequence alignments rather than a single sequence. The sampled posterior decoding performs better than the exact posterior, which in turn performs better than the Viterbi.

Key performance measures

The first column in Table 1 (SOV) is the primary key performance measure employed within
Table 1. The accuracy of predictions as measured by standard performance measures: SOV on 3-states, Q3, Q13. The highest accuracy in each column is shown in bold, and the standard error of the mean is shown after each number.

|          | SOV (%) | Q3 (%) | Q13 (%) |
|----------|---------|--------|---------|
| Alignment|         |        |         |
| Profile only | 81.2 ± 0.2 | 77.3 ± 0.2 | 55.8 ± 0.3 |
| Exact Viterbi | 79.3 ± 0.2 | 75.3 ± 0.2 | 53.4 ± 0.3 |
| Exact post. | 80.5 ± 0.2 | 76.2 ± 0.2 | 54.2 ± 0.3 |
| Sampled post. | 83.0 ± 0.2 | 77.4 ± 0.2 | 55.2 ± 0.3 |
| Single sequence |         |        |         |
| Profile only | 71.3 ± 0.2 | 65.8 ± 0.2 | 45.3 ± 0.3 |
| Exact Viterbi | 72.4 ± 0.2 | 64.5 ± 0.2 | 43.3 ± 0.3 |
| Exact post. | 73.8 ± 0.2 | 65.4 ± 0.2 | 43.3 ± 0.2 |
| Sampled post. | 75.3 ± 0.2 | 66.3 ± 0.2 | 44.2 ± 0.2 |

Table 2. Quality of predictions. X is the profile and M is the joint profile + k-mer model. The probabilities are reported per residue; that is, the quantity shown is \( \prod_i P(y_i)^1/L \), where the product is over all real sequences \( y \) in the test set and \( L \) is the sum of their lengths.

|          | \( P(\text{real seq}|X) \) | \( P(\text{real seq}|M) \) |
|----------|-----------------------------|-----------------------------|
| Alignment| 0.271                       | 0.385                       |
| Single sequence | 0.189                       | 0.325                       |

Fig. 3. Distribution of scores for samples from a profile and from the corresponding joint model. Each dot represents a sequence. The axes are the two components of the joint model \( M \). The red cloud (bottom) represents 50,000 samples from the profile \( X \); the blue cloud (top) represents 50,000 samples from the joint model \( M \); circle is the real sequence. The profile used is the same as in Fig. 2.

Confidence scores The key performance measures above are commonly used in the field but fail to measure a crucial aspect of the prediction: almost as valuable as the predicted secondary structure states, for practical applications, is knowing the confidence of the predictions. A good way of measuring the overall prediction quality, taking into account the accuracy of the confidence at each position, is to calculate the probability of emitting the correct sequence.

Table 2 reports the probabilities per residue of observing the real sequence being emitted from the profile and from the joint profile + k-mer model respectively. It can be seen that, for both single-sequence and alignment inputs, the odds of observing the real sequence increase dramatically in the joint model. This improvement is substantial and represents a major new contribution of this work. It is therefore worth examining in more detail and we do so in Fig. 3, which shows sampling from the model using the same profile that we show in the example in Fig. 2. Sampling sequences from the original profile produces the red cloud. This cloud has largely negative k-mer scores (vertical axis) highlighting an un-protein-like characteristic of the sequences. By combining the k-mer model with the profile, the red sequences are heavily penalised and would no longer be likely to be sampled. The k-mer model restricts the space from which sequences can be sampled and what remains is the blue cloud, which is far more likely to contain the real sequence (circle).
Table 3. The five most encouraged and discouraged k-mers for multiple alignments.

| Encouraged k-mers | Discouraged k-mers |
|-------------------|-------------------|
| k-mer             | Mean score        | k-mer             | Mean score        |
| MM                | 3.4               | MA                | -13.0             |
| EE                | 3.0               | PM                | -11.7             |
| ZE                | 2.2               | HY                | -11.4             |
| GG                | 2.1               | GY                | -11.2             |
| YZ                | 1.9               | TP                | -11.1             |
| PMS               | 5.7               | CIZ               | -9.1              |
| HQE               | 5.1               | QEZ               | -9.0              |
| TMA               | 4.8               | YTC               | -8.9              |
| YQY               | 4.7               | CTC               | -8.3              |
| ZQZ               | 4.1               | ZEQ               | -8.3              |
| YEQY              | 7.6               | CGGC              | -7.5              |
| HQBB              | 7.2               | CGGS              | -7.0              |
| QEZM              | 6.9               | CGGT              | -7.0              |
| QBBQ              | 6.8               | CHHT              | -6.6              |
| BTQM              | 6.6               | CGGH              | -6.3              |

Convergence of the training procedure. For simplicity we will restrict our discussion to profiles built from alignments, as the behaviour for profiles built from single sequences is similar. For sequences sampled straight from the profile, which is the zeroth iteration of our procedure, the distribution of 1-mers is very close to that in the training set, with the Kullback-Leibler divergence (15) ranging from $D_1^{(0)} = 8 \times 10^{-5}$ to $6 \times 10^{-4}$ for the three training sets. On the other hand the 2- to 4-mer distributions are very different and get progressively worse as $k$ increases, from $D_2^{(0)} \approx 4 \times 10^{-4}$ to $D_4^{(0)} \sim 1$. This is expected behaviour, because the neural network is essentially trained on 1-mer accuracy. After the first iteration the 1-mers worsen to $D_1^{(1)} \sim 1 \times 10^{-2}$, but 2- to 4-mers improve to $D_2^{(1)} \sim 2 \times 10^{-2}$ and $D_4^{(1)} \sim 5 \times 10^{-2}$. After a total 15 iterations, the final divergences for models used in the rest of the results section are as follows: $D_1^{(15)} \sim 5 \times 10^{-4}$, $D_2^{(15)} \sim 7 \times 10^{-4}$ and $D_4^{(15)} \sim 1 \times 10^{-3}$. For comparison, the divergences among the three training sets are $D_1^{T-T} \sim 4 \times 10^{-4}$, $D_2^{T-T} \sim 8 \times 10^{-4}$ and $D_4^{T-T} \sim 1 \times 10^{-2}$.

4 DISCUSSION OF THE METHOD

The method we present here can be thought of as a graphical model. The formal structure is that of a Dynamic Conditional Random Field (CRF) (Lafferty et al., 2001; Rohanimanesh et al., 2007). Although our model (10) is a CRF, compared to usual practice in the field there is a major difference, which is our simple training algorithm (13). The algorithm was inspired by our reformulation of a Markov chain with memory as a hierarchical model of k-mers, where the k-mer scores are log-ratios of the observed frequency relative to the expected frequency based on the k – 1 level of the model. The appearance of log-odds scores is particularly exciting, because they underlie much of sequence alignment theory, including statistical assessment of alignment significance.

The conclusion during recent rounds of CASP for tertiary structure prediction has been to try many potential alignments and secondary structure predictions and to defer judgement until a full 3D model has been built, and to assess that model. In profile-profile alignment (Madera, 2008; Soeding, 2005; Sadreyev and Grishin, 2003), likewise, one is not interested in the single best sequence, but rather in a large number of samples of plausible ones. For this reason we argue that the $P(\text{real seq}|M)$ quality measure is more important than the SOV, Q or Q13 measures, although these remain the most popular in the field. The $P(\text{real seq}|M)$ measure requires confidences to be assigned to the sequence at each position and rewards for accurate confidence as well as correct prediction; a best guess is far less useful without knowledge of which parts to trust. Calculation of the partition function (11) is not needed for sampling and majority voting, i.e. for almost any practical application, but we did this for the purposes of assessing $P(\text{real seq}|M)$; for more complex future incarnations of the model this in turn may need to be handled using sampling methods (Wang and Landau, 2001). Another issue affecting future extensions of the model is that sampling is currently slow. In our simple 4-mer model the most accepted mutations lie on the ends of helices and sheets, either extending or shortening them by one residue. Medium and long-range models are likely to further slow the sampling process and create lock-ins due to very long-range repulsive interactions between parallel and anti-parallel sheets, so approaches which avoid local minima will need to be explored, e.g. parallel tempering (Earl and Deem, 2005).

To apply our framework to medium-range and long-range interactions we need a hierarchical model of whole helices, loops, parallel, anti-parallel and mixed strands. We can sample the distribution of real secondary structure lengths and correlations between neighbouring element lengths, giving us k-mer scores on an alphabet of whole secondary structure elements. We have already solved the correlated null model (though this is equivalent to Markov model of order $n-1$), for STR2 k-mers by sampling realistic sequences from a profile. This can be used for alignment, the next step being to generate a pairwise scoring function which would be a mix of the traditional substitution matrix and our sequence model. Not many people are using extended secondary structure alphabets. Clearly the richer alphabets contain more information, and in many cases, whether profile-profile, or using multi-track models, the more information the better. It is likely that the reason more advances have not been seen in homology recognition due to the addition of secondary structure is that we're not using the information correctly.

In 3-fold validation of the training, the mean deviation of scores for the k-mers between sets was relatively low with a value of 0.479, i.e. on average, scores will not be further than half the mean score across sets. This indicates that training on each subset resulted in convergence to similar distributions.
5 CONCLUSION

We have succeeded in producing a new method which is an additional layer on top of existing neural network-based secondary structure prediction methods meaning that any improvements we make, de facto, represent an advance on the state of the art. Our method has succeeded in our goal of sampling more realistic secondary structure sequences from a profile without loss of accuracy, in fact we have surpassed this goal and actually increased the prediction performance. We have managed to significantly increase on the SOV scores (+1.8%) and there is no significant difference in the less sophisticated Q3 scores (+0.1%) which are the two industry standard measures, e.g. used in CASP. A more important measure of the quality of predictions, however, is \( P(\text{real seq}|M) \), the probability of sampling the correct sequence from the model; this takes into account the confidence scores for each position, essential for practical applications using the prediction. We dramatically improve this probability from 0.271 to 0.385. The \( k \)-mer model we demonstrated here only goes up to strings of length 4, which is well inside the \(+/-3\) residue window of the neural network so there is no new information provided to the model on top of what the neural network could see. There is still great potential for further improvement in the future by extending this approach in different ways: most simply with longer \( k \)-mers, but also by creating an alphabet of whole secondary structure elements. Data indicates that this will especially improve predictions for parallel / anti-parallel sheets. Although we demonstrated our method on a specific neural network, \( k \)-mer models can be trained to correct the emissions of any other neural networks for secondary structure prediction. The work we present here not only improves on secondary structure prediction, but our theoretical framework for modelling higher order interactions in proteins opens up a way forward for the advancement of protein sequence analysis in general.

ACKNOWLEDGEMENT

Funding: This work was supported by the European Commission 7th framework programme [grant number 213037].

REFERENCES

Aloby,P., Strock,A., Hadley,C. and Russell,R.B. (2003) Predictions without templates: new folds, secondary structure, and contacts in CASP5. Proteins: Struct., Funct., Genet., 53, 436-456.

Bystroff,C., Thorsson,V. and Baker,D. (2000) HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins. J. Mol. Biol., 301, 173-190.

Casella,G. and George,E.I. (1992) Explaining the Gibbs Sampler. Amer. Statistician, 46, 167-174.

Cuff,J.A., Clamp,M.E., Siddiqui,A.S., Finlay,M. and Barton,G.J. (1998) IPred: a consensus secondary structure prediction server. Bioinformatics, 14, 892-893.

Do,C.B., Woods,D.A. and Batsoglou,S. (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. Bioinformatics, 22, e90-e98.

Durbin,R., Eddy,S.R., Krogh,A. and Mitchison,G. (1998) Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge.

Earl,D.J. and Deem,M.W. (2005) Parallel tempering: Theory, applications, and new perspectives. Phys. Chem. Chem. Phys., 7, 3910-3916.

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

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

Karchin,R., Cline,M., Mandel-Gutfreund,Y. and Karplus,K. (2003) Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. Proteins: Struct., Funct., Genet., 51, 504-514.

Karlof,K. (2009) SAM-T08, HMM-based protein structure prediction. Nucleic Acids Res., 37, W492-W497.

Kratzmann,S., Barrett,C., Thiligen,G., Karchin,R. and Karplus,K. (2008) PREDICT-2ND: a tool for generalized protein local structure prediction. Bioinformatics, 24, 2453-2459.

Koh,J.Y.Y., Volker,VA., Marti-Renom,M.A., Przybylski,D., Madhusudhan,M.S., Eswar,N., Grana,O., Papos,F., Valencia,A., Salt.A. and Rost,B. (2003) EVA: evaluation of protein prediction servers. Nucleic Acids Res., 31, 3311-3315.

Lafferty,J., McCallum,A. and Pereira,F. (2001) Conditional random fields. Probabilistic models for segmenting and labeling sequence data. In Proc. 16th Int. Conf. on Machine Learning, 282-289.

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.

Madera,M. (2008) Profile Comparer: a program for scoring and aligning profile hidden Markov models. Bioinformatics, 24, 2630-2631.

Madera,M. and Gough,J. (2002) A comparison of hidden Markov model procedures for remote homology detection. Nucleic Acids Res., 30, 4321-8.

Metropolis,N., Rosenbluth,A.W., Rosenbluth,M.N. and Teller,A.H. (1953) Equation of state calculations by fast computing machines. J. Chem. Phys., 21, 1087-1092.

Moult,J., Pedersen,J.T., Judson,R. and Fidelis,K. (1995) A large-scale experiment to assess protein structure prediction methods. Proteins, 23, ii-v.

Ouali,M. and King,R.D. (2000) Cascaded multiple classifiers for secondary structure prediction. Protein Sci., 9, 1162-1176.

Pollastri,G. and McLysaght,A. (2005) PORTER: a new, accurate server for protein secondary structure prediction. Bioinformatics, 21, 1719-1720.

Rohanimanesh,K., McCallum,A. and Sutton,C. (2007) Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. J. Mach. Learn. Res., 6, 693-723.

Sadreyev,R.I. and Grishin,N.V. (2003) COMPASS: a tool for comparison of multiple protein alignments with assessment of statistical significance. J. Mol. Biol., 326, 317-316.

Sato,K. and Sakakibara,Y. (2005) RNA secondary structural alignment with conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. J. Mol. Biol., 357, 232-242.

Sieding,J. (2005) Protein homology detection by HMM-HMM comparison. Bioinformatics, 21, 951-960.

Wang,G. and Dunbrack,R.L. (2003) PISCES: a protein sequence culling server. Bioinformatics, 19, 1589-1591.

Wang,F. and Landau,D.P. (2001) Efficient, multiple-range random walk algorithm to calculate the density of states. Phys. Rev. Lett., 86, 2050-2053.

Won,K.J., Hamelryck,T., Prugel-Bennett,A. and Krogh,A. (1999) An evolutionary method for learning HMM structure: prediction of protein secondary structure. BMC Bioinformatics, 8, 357.

Zemka,A., Venkovs,C. and Rost,B. (1999) A modified definition of Sos, a segment-based measure for protein secondary structure prediction assessment. Proteins, 34, 220-223.