BIOINFORMATICS ORIGINAL PAPER Vol. 26 no. 5 2010, pages 596–602
doi:10.1093/bioinformatics/btq020

Sequence analysis

Advance Access publication February 3, 2010

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, University of Bristol, Woodland Road, Bristol BS8 1UB, UK and [2]Department of Biomolecular Engineering, University of California, Santa Cruz, CA 95064, USA

Associate Editor: Burkhard Rost

ABSTRACT

Motivation: Some first order methods for protein sequence analysis inherently treat each position as 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 1. 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 an 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 Segment OVerlap (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

Received on September 21, 2009; revised on December 24, 2009; accepted on January 12, 2010

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 *et al*., 1995) competition for protein structure prediction ceased to assess this as an official category some years ago, as has the EVA (Koh *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 utilized 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. Cuff *et al*., 1998; Jones, 1999; Katzman *et al*., 2008; Ouali and King, 2000; Pollastri and McLysaght, 2005; Rost, 1996), all using neural networks. Analysis of the results of the last CASP competition to include secondary structure (Aloy *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 *et al*., 2008), which is named as one of the three 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 article 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.

The idea of implicitly using more than one amino acid has been around for some time (Nagano, 1973; Chou and Fasman, 1978), but many protein sequence comparison methods, when making a prediction, implicitly treat positions in a protein sequence with some level of independence. Even in cases where a sliding window is used, predictions are dependent on the neighbouring amino acids, but not usually directly on neighbouring predictions. While an 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 α-helix, the adjacent positions are highly likely ($\sim$90% chance) to also be α-helical. In fact, we have observed three broad types of correlations that violate the independence assumption:

(1) *Short range*: if `H` stands for a helix and `A` for an anti-parallel strand, there are three times more occurrences of `HH`, 14 times more occurrences of `AA`, but at least 10 000 times fewer

*To whom correspondence should be addressed.

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 article, 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 Section 4 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 (1mer), each possible pair (2mer) 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-predicted compared with real sequences, and penalize k-mers that are over-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 (Do *et al.*, 2006; Liu *et al.*, 2004; Sato and Sakakibara, 2005) 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 first-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 reformulation 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 subsequence of y be denoted $y_{m\ldots n}$. Let us now suppose that sequences in some large training dataset $\mathcal{T}$ can be modelled as Markov chains of order $n-1$, i.e. that

$$P(y)=P(y_{1\ldots n-1})P(y_n|y_{1\ldots n-1})P(y_{n+1}|y_{2\ldots n})\ldots P(y_L|y_{L-n+1\ldots L-1}). \quad (1)$$

We can express the individual probabilities in (1) in terms of the distribution of k-mers in $\mathcal{T}$. Let us denote the relative frequency of a k-mer a in $\mathcal{T}$ 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

$$P(y_{1\ldots n-1}|T)=T_{n-1}(y_{1\ldots n-1}), \quad (2)$$

and the transition probabilities are

$$P(y_m|y_{m-n+1\ldots m-1},T)=\frac{T_n(y_{m-n+1\ldots m})}{T_{n-1}(y_{m-n+1\ldots m-1})}. \quad (3)$$

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:

$$\ln P(y|T)=\mathbf{S}\cdot\mathbf{F}(y), \quad (4)$$

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

$$S_1(a_1|T) = \ln T_1(a_1) \quad (5)$$

$$S_2(a_{1\ldots 2}|T) = \ln \frac{T_2(a_{1\ldots 2})}{T_1(a_1)T_1(a_2)} \quad (6)$$

$$S_3(a_{1\ldots 3}|T) = \ln \frac{T_3(a_{1\ldots 3})}{\frac{T_2(a_{1\ldots 2})T_2(a_{2\ldots 3})}{T_1(a_2)}}, \quad (7)$$

and so on up to n-mers. An important aspect of this formulation is that Equations (6) and (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

$$\boxed{S_k(a_{1\ldots k}|T)=\ln \frac{T_k(a_{1\ldots k})}{P(a_{1\ldots k}|T_{k-1})}}. \quad (8)$$

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

$$P(y|X)=\prod_{l=1}^{L}P\left(y_l|X_l\right). \quad (9)$$

Our approach is to modify this emission probability by introducing a joint profile + k-mer model M,

$$P(y|M)=\frac{1}{Z(X,R)}\exp\left\{\ln P(y|X)+\mathbf{R}\cdot\mathbf{F}(y)\right\}, \quad (10)$$

where

$$Z(X,R)=\sum_{y'}\exp\left\{\ln P(y'|X)+\mathbf{R}\cdot\mathbf{F}(y')\right\} \quad (11)$$

is the normalization factor (also called the partition function) and $\mathbf{R}$ 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:

$$R_k^{(0)}(a) = 0 \quad (12)$$

$$R_k^{(i)}(a) = R_k^{(i-1)}(a)+S_k(a|T)-S_k(a|B^{(i-1)}). \quad (13)$$

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 $\mathcal{B}^{(i-1)}$ sampled from iteration $i-1$ of the model.

The S_k scores in (13) are undefined when a is absent from $\mathcal{T}$ or $\mathcal{B}^{(i-1)}$. 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 $\mathcal{T}$ but present in $\mathcal{B}^{(i-1)}$), we reset $T_k(a)$ to satisfy

$$S_k(a|T) - S_k(a|B^{(i-1)}) = 0. \tag{14}$$

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 $\mathcal{T}$, we further reset it to the cut-off 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 $\mathcal{T}$ 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,

$$D_k^{(i)} = \sum_{a_{1\ldots k}} T_k(a) \log_2 \frac{T_k(a)}{B_k^{(i)}(a)}. \tag{15}$$

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 $T \log \frac{T}{B}$ score is taken to be zero and both $T_k(a)$ and $B_k^{(i)}(a)$ 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 $B_k^{(i)}$ are renormalized.

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,

$$\hat{y}^{Vit} = \underset{y}{\text{argmax}} P(y|M), \tag{16}$$

or the forward–backward algorithm to perform posterior decoding (also known as marginalization), which for each position i computes the letter $\hat{y}_i^{\text{post}}$ most likely observed at that position,

$$\hat{y}_i^{\text{post}} = \underset{y_i}{\text{argmax}} \sum_{y_{1\ldots i-1}} \sum_{y_{i+1\ldots L}} P(y|M). \tag{17}$$

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 Markov Chain Monte Carlo (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 (Casella and George, 1992).

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 $\hat{y}^{\text{post}}$. Instead, we can calculate an estimate $y_d^{\dagger}$ of the posterior decoding by analysing a set of d samples drawn from the distribution $P(y|M)$. In this case, $\left(y_d^{\dagger}\right)_i$ is simply the classification most frequently observed at position i across the sample set. As d tends to infinity, $y_d^{\dagger}$ becomes equivalent to $\hat{y}^{\text{post}}$.

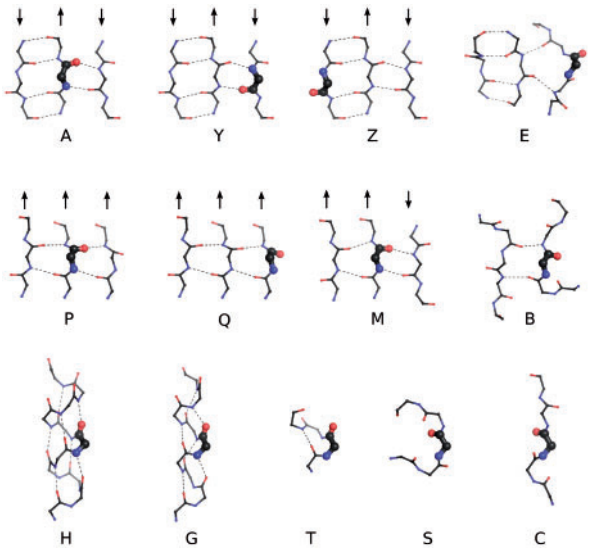

Fig. 1. The STR2 alphabet. This 13-state alphabet uses DSSP hydrogen bond definitions and is defined strictly from DSSP output. The main difference is that STR2 subdivides the DSSP class E (β-sheet) into seven classes: A M P, anti-parallel, mixed or parallel β-strand, hydrogen bonded to two partners; Y Z, anti-parallel edge strand residue, bonded and non-bonded, respectively; Q, parallel edge strand, both bonded and non-bonded residues; and E, all other β-sheet residues, typically β-bulges. STR2 groups together DSSP classes H (α-helix) and I (π-helix) into a single STR2 class H. The remaining five classes are identical to DSSP: G, 3_{10} helix; T, turn; S, bend; C, coil; and B, β-bridge.

2.5 The STR2 alphabet

The UCSC STR alphabet, described in Figure 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 (Karchin *et al*., 2003). Possibly for this reason, it has been the most successful alphabet at UCSC in protein alignment and fold recognition tests.

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-30pc-1763*, created by 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 3-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 while 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 four-layered (others use two layers) feed-forward networks taking as input a sliding window of 27 residues worth of multiple alignment profile information (i.e. 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 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 utilized are available at http://www.soe.ucsc.edu/~karplus/predict-2nd/ and the SAM-T08 (Karplus, 2009) web site.

3 RESULTS

Overview: In Figure 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 Figure 2, the first strand of the sequence within the profile is evidently incorrect (at the very top) with respect to the true 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.

Tables 1 and 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 normalized for sequence length.

We observe that while 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 Segment OVerlap (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 CASP (Moult *et al*., 1995). SOV in its current form was defined

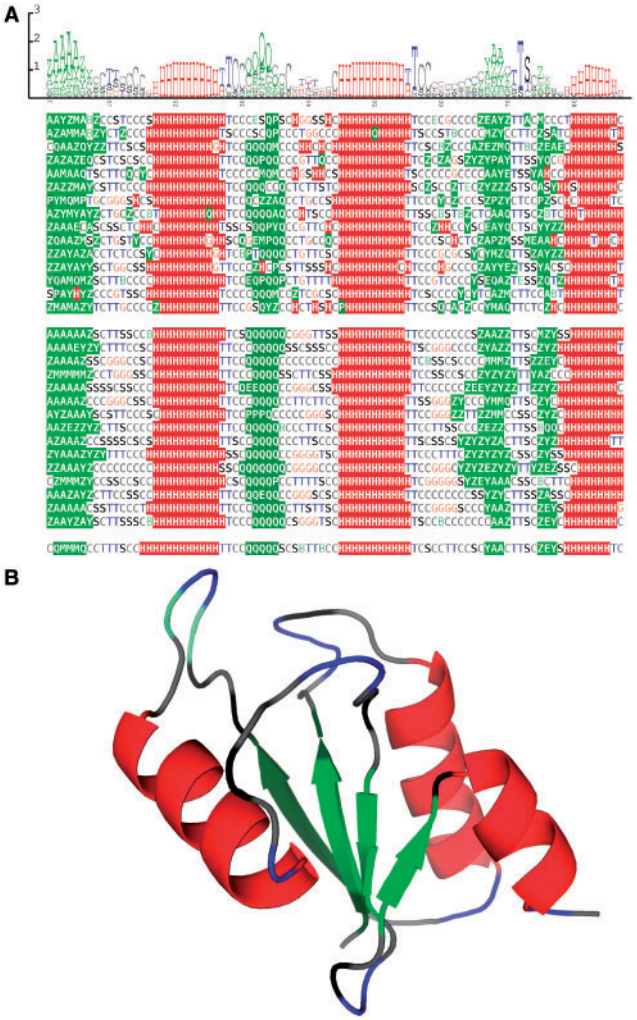

Fig. 2. Improvement due to k-mer model. (**A**) The improvements can be seen by comparing the two blocks of secondary structure sequences: above are the results from sampling columns independently and below are results from correlated sampling using the k-mer model (10). The STR2 profile is shown graphically above the alignments, and the true secondary structure is shown at the bottom, and in (**B**) which has the same colouring scheme showing the elements on the PDB structure (1aba). N.B. The quality of individual rows is important, not the alignment.

by Zemla *et al*. (1999) and is a segment-oriented definition of prediction accuracy measured as a percentage (calculated on the 3-letter alphabet). Also used in CASP is Q_3 (Rost and Sander, 1993), which is simpler per-residue measure of percentage prediction accuracy for the standard 3-letter EHL secondary structure alphabet. We generalize the definition of Q to the STR2 alphabet to produce the third column, Q_{13}, where 13 the size of the STR2 alphabet.

Historically, Q_3 has been an important measure of secondary prediction accuracy and so is included for reference, although evidently predicting true sequences under the 13-letter STR2 alphabet is a far more difficult problem. To calculate Q_3, we translated sequences from their 13-state representation to the EHL alphabet using the mapping: CST → L, HG → H, AYZMPQBE → E.

Table 1. The accuracy of predictions as measured by standard performance measures: SOV on 3-states, Q3, Q13

		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	43.1 ± 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

The highest accuracy in each column is shown in bold, and the standard error of the mean is shown after each number.

Table 2. Quality of predictions

	P(real seq|X)	P(real seq|M)
Alignment	0.271	**0.385**
Single sequence	0.189	**0.325**

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 $[\prod P(y)]^{1/L}$, where the product is over all real sequences y in the test set and L is the sum of their lengths.

It can be seen that posterior sampling from our k-mer model on alignment inputs produces a superior SOV accuracy score to all other input types and decoding methods (83%), and that performance is not significantly different for the Q_3 measure (77.4%). SOV and Q_3 results are, respectively, 1.8% and 0.1% better than those produced directly from a profile, and this rises to 4% and 1.5% when using single sequence rather than alignment inputs. The harder Q_{13} measure actually shows a small decrease in performance (55.8 to 55.2% and 45.3 to 44.2% for alignment and single-sequence inputs, respectively).

The difference observed between the sampled and exact posterior is due to undersampling; if you sample sufficiently this difference goes away. Furthermore, if we did the training on the exact posterior instead of the sampled posterior, we would expect the difference to reverse, with the exact posterior improving on the current sampled posterior (although perhaps not noticeably).

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 Figure 3, which shows sampling from the model using the same profile that we show in the example in Figure 2. Sampling sequences from the original profile produces the red cloud. This cloud has largely negative k-mer scores (vertical axis) highlighting an unprotein-like characteristic of the sequences. By combining the k-mer model with the profile, the red sequences are heavily penalized 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).

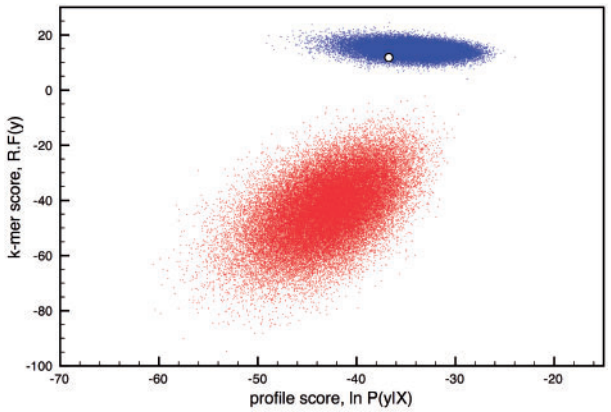

Fig. 3. Distribution of scores for samples from profile and corresponding joint model. Each dot represents a sequence. The axes are the two components of the joint model M. The red cloud (bottom) represents 50 000 samples from the profile X; the blue cloud (top) represents 50 000 samples from the joint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

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	CTZ	−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

Notable characteristics of the output k*-mer distributions* Table 3 shows geometric averages over the three training sets for alignment inputs for some of the most encouraged and discouraged k-mers. These reflect structural expectations, e.g. the five most discouraged 4mers all feature two-residue helices (none of which is possible in the real world) and QEZ and ZEQ feature among the

most discouraged 3mers due to the necessity of maintaining the parallel/anti-parallel nature of a strand across a beta-bulge. Other 3mer observations include: all single-residue helices are discouraged; so are single-residue edge strands; and single-residue turns except immediately after or right before a helix, and same for the reverse transitions from parallel or anti-parallel to mixed. The 4mer observations reinforce some 3mer observations but also include: 3-turns (characterized by i to $i+3$ hydrogen bonds) are unlikely to occur except when adjacent to helices or edge β-strands; as with β-bulges, strands are more likely to be contiguous than broken; and double-partner β-strand residues `A` and `P` are unlikely to occur in runs of less than three. The most encouraged 3- and 4mers are so rare that they are of no importance and most of them are artefacts, e.g. `PM` occurs only once in the training set and it happens to be `PMS`.

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.

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 1mers is very close to that in the training set, with the Kullback–Leibler divergence (15) ranging from $D_1^{(0)} = 8\times10^{-5}$ to 6×10^{-4} for the three training sets. On the other hand, the 2- to 4mer distributions are very different and get progressively worse as k increases, from $D_2^{(0)} \sim 4\times10^{-1}$ to $D_4^{(0)} \sim 1$. This is expected behaviour, because the neural network is essentially trained on 1mer accuracy. After the first iteration, the 1mers worsen to $D_1^{(1)} \sim 1\times10^{-2}$, but 2- to 4mers improve to $D_2^{(1)} \sim 2\times10^{-2}$ and $D_2^{(1)} \sim 5\times10^{-2}$. After a total 15 iterations, the final divergences for models used in the rest of this section are as follows: $D_1^{(15)} \sim 5\times10^{-4}$, $D_2^{(15)} \sim 7\times10^{-4}$ and $D_4^{(15)} \sim 1\times10^{-3}$. For comparison, the divergences among the three training sets are $D_1^{T-T} \sim 4\times10^{-4}$, $D_2^{T-T} \sim 8\times10^{-4}$ and $D_4^{T-T} \sim 1\times10^{-2}$.

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 CRF (Lafferty *et al*., 2001; Rohanimanesh *et al*., 2007). Although our model (10) is a CRF, compared with 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; Sadreyev and Grishin, 2003; Soeding, 2005), 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_3 or Q_{13} 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 4mer 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- 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 are not using the information correctly.

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 Q_3 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. Using the K-mer model, we have demonstrated that when $K=4$ we can

gain the above improvements. Strings of length 4 are well inside the +/− 13 residue window of the neural network, so the improvements are achieved without using any new information beyond what the neural networks are already using. 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 also our theoretical framework for modelling higher order interactions in proteins opens up a way forward for the advancement of protein sequence analysis in general.

Funding: European Commission 7th framework programme (grant number 213037).

Conflict of Interest: none declared.

REFERENCES

Aloy,P. *et al.* (2003) Predictions without templates: new folds, secondary structure, and contacts in CASP5. *Proteins Struct., Funct. Genet.*, **53**, 436–456.

Bystroff,C. *et al.* (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. *Am. Stat.*, **46**, 167–174.

Chou,P.Y. and Fasman,G.D. (1978) Prediction of the secondary structure of proteins from their amino acid sequence. *Adv. Enzymol. Relat. Areas Mol. Biol.*, **47**, 45–148.

Cuff,J.A. *et al.* (1998) JPred: a consensus secondary structure prediction server. *Bioinformatics*, **14**, 892–893.

Do,C.B. *et al.* (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics*, **22**, e90–e98.

Durbin,R. *et al.* (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. *et al.* (2003) Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. *Proteins Struct. Funct. Genet.*, **51**, 504–514.

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

Katzman,S. *et al.* (2008) PREDICT-2ND: a tool for generalized protein local structure prediction. *Bioinformatics*, **24**, 2453–2459.

Koh,I.Y.Y. *et al.* (2003) EVA: evaluation of protein prediction servers. *Nucleic Acids Res.* **31**, 3311–3315.

Lafferty,J. *et al.* (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, Morgan Kaufmann, CA, USA, pp. 282–289.

Liu,Y. *et al.* (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**, 4321–4328.

Metropolis,N. *et al.* (1953) Equation of state calculations by fgast computing machines. *J. Chem. Phys.*, **21**, 1087–1092.

Moult,J. *et al.* (1995) A large-scale experiment to assess protein structure prediction methods. *Proteins*, **23,** ii–v.

Nagano,K. (1973) Logical analysis of the mechanism of protein folding. I. Prediction of helices, loops and β-structures from primary structure. *J. Mol. Biol.*, **75**, 401–420.

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. *et al.* (2007) Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. *J. Mach. Learn. Res.*, 8, 693–723.

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

Rost,B. and Sander,C. (1993) Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.*, **232**, 584–599.

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**, 317–336.

Sato,K. and Sakakibara,Y. (2005) RNA secondary structural alignment with conditional random fields. *Bioinformatics*, **21**, ii237–ii242.

Soeding,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. *et al.* (2007) An evolutionary method for learning HMM structure: prediction of protein secondary structure. *BMC Bioinformatics*, **8**, 357.

Zemla,A. *et al.* (1999) A modified definition of SOV, a segment-based measure for protein secondary structure prediction assessment. *Proteins*, **34**, 220–223.