Protein Secondary Structure Prediction with Bidirectional Recurrent Neural Nets: can weight updating for each residue enhance performance?

Michalis Agathocleous[1], *Georgia Christodoulou*[1], *Vasilis Promponas*[2], *Chris Christodoulou*[1⋆], *Vassilis Vassiliades*[1], and *Antonis Antoniou*[1]

Dept. of Computer Science[1], Dept. of Biological Sciences[2], University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus.
michalis.agathocleous.09@ucl.ac.uk,cs06gc1@cs.ucy.ac.cy, vprobon@ucy.ac.cy,cchrist@cs.ucy.ac.cy v.vassiliades@cs.ucy.ac.cy,a.antoniou@cs.ucy.ac.cy

Abstract. Successful protein secondary structure prediction is an important step towards modelling protein 3D structure, with several practical applications. Even though in the last four decades several PSSP algorithms have been proposed, we are far from being accurate. The Bidirectional Recurrent Neural Network (BRNN) architecture of Baldi et al. [1] is currently considered as one of the optimal computational neural network type architectures for addressing the problem. In this paper, we implement the same BRNN architecture, but we use a modified training procedure. More specifically, our aim is to identify the effect of the contribution of local versus global information, by varying the length of the segment on which the Recurrent Neural Networks operate for each residue position considered. For training the network, the backpropagation learning algorithm with an online training procedure is used, where the weight updates occur for every amino acid, as opposed to Baldi et al. [1], where the weight updates are applied after the presentation of the entire protein. Our results with a single BRNN are better than Baldi et al. [1] by three percentage points (Q3) and comparable to results of [1] when they use an ensemble of 6 BRNNs. In addition, our results improve even further when sequence-to-structure output is filtered in a post-processing step, with a novel Hidden Markov Model-based approach.

Key words: Protein Secondary Structure Prediction, Bidirectional Recurrent Neural Networks, Bioinformatics and Computational Biology

1 Introduction

Proteins are linear polymers of amino acids, and their complement in any living cell provides a vast repertoire of functions required for maintaining life. Their functionality is determined by the detailed three dimensional arrangement of their constituent atoms, which specifies the shape of the molecule and the ways

⋆ Corresponding Author

it can potentially interact with other biomolecules. Biochemical methods for experimental determination of protein tertiary structure are expensive, time consuming and frequently ineffective. On the contrary, modern high-throughput molecular biology techniques provide us daily with an increasing number of (putative) amino acid sequences, corresponding to proteins of unknown structure and function. It is widely accepted that protein sequence determines its 3D structure [2]; however, the exact mapping from the 1D to the 3D space has not yet been elucidated. Nevertheless, knowledge of local regular arrangements of amino acid residues (secondary structural elements), may indicate the structure adopted by a protein chain and provide useful constraints for further structural modelling.

The protein secondary structure prediction (PSSP) problem has been addressed with different computational approaches during the last four decades. Mainly due to the lack of necessary volumes of structural data, only relatively recently were machine learning methods (including artificial neural networks - NNs) introduced to this field. Such methods have resulted in considerably higher predictive performance compared to preceding empirical or statistical methods, with per residue accuracies (Q3 [3]) ranging from 63% to 76% [4], [5]. In particular, a fully connected feed forward NN and a single hidden layer, with a local input window of 13 amino acids using orthogonal (or one-hot) encoding has been applied by Qian and Sejnowski [6] reaching accuracy performance of Q3 = 64.3%. Based upon this work, Rost and Sander developed the so–called PHD method [7], [8], which used various machine learning methods including early stopping and ensemble of averages of different NNs. Their most substantial improvement was, however, the use of multiple sequence alignments (MSA) that enrich the input with evolutionary information [8]. Through their work the accuracy performance reached to Q3 = 74%. Another interesting approach, similar to PHD, is the work of Salamov and Soloveyev [9] that introduced a variant of the nearest-neighbor approach that could achieve similar accuracy (Q3 = 73.5%) using a single sequence as input. Cuff and Barton [10] built another MSA-enabled feed forward NN (of a larger size however) with comparable accuracy.

A major breakthrough in the field of PSSP is considered to be the work of Baldi and colleagues [1], who introduced a Bidirectional Recurrent Neural Network (BRNN). The main motivation of this work was the inability of feed forward NNs with a fixed-width sliding window to capture long-range dependencies. The aforementioned BRNN takes as input a fixed-width window centered each time at the residue of interest. However, it attempts to predict the secondary structure for the target residue by considering not only the local context (i.e., neighboring residues in the window) but also residues located on the left and right of the target residue [1]. These bidirectional dynamics proved to be able to capture both upstream and downstream information. When the BRNN processed evolutionary information in the form of MSAs, the predictive performance reached up to Q3 = 73.6% [1], which was increased to 75.1% on average, when an ensemble of 6 predictors was used. Along the same lines Chen and Chaudhari [11] developed a cascaded architecture, consisting of two BRNNs where the second network fil-

ters the output of the first. As they report, their system achieves 74.38% (Q3) accuracy with a Segment Overlap (SOV [12]) score of approximately 66%.

In the introductory work of BRNNs for PSSP [1], for each position in a sequence presented to the network an input window is formed centered around this position. During the training phase, the protein is processed in its entirety before the weight updates are made. In this paper, our aim is to investigate how the prediction accuracy could be improved by: (i) updating the weights at every residue, which in a way constitutes a form of dynamic training and is more context-sensitive, and (ii) using different filtering approaches (a novel method based on Hidden Markov Models and a cascaded feed forward Artifical NN).

The rest of the paper is organised as follows: section 2 presents the methodology followed by the results and discussion in section 3; section 4 gives the conclusions and future work.

2 Methods

2.1 Data Collection and Preprocessing

Sequence Similarity-Based Redundancy Reduction: In order to train and validate the BRNN, we need a set of high quality data consisting of proteins with experimentally determined 3D structures deposited in the RSCB Protein Data Bank (PDB) (URL: http://www.pdb.org/, accessed 20 April 2009). Moreover, the resulting dataset should be maximal (in order to capture the knowledge we currently have available on protein structures) but also non-redundant, to avoid poor generalisation of the BRNN. For this purpose we utilised the PDB-Select25 dataset (URL: http://bioinfo.tg.fh-giessen.de/pdbselect/, accessed 20 April 2009), which is regularly produced by analysis of the PDB with the algorithm described in [13]. This dataset contained 4019 polypeptide chains that shared less than 25% overall pair-wise sequence identity.

Data Selection Criteria: It is of great importance to choose a suitable data set so that the BRNN can be properly trained, and several criteria should be fulfilled for the selection process:

- We retained only entries determined by X-ray diffraction, for which we can apply the resolution as a quantitative selection measure. In particular, a 3.0 Å threshold was used to discard structures of insufficient/questionable quality.
- We discarded entries with physical chain breaks, as empirically identified by at least one pair of successive C_α atoms with a distance longer than 4.0 Å, as well as proteins with large segments of undefined secondary structure.
- Chains with a length of less than 30 amino acids were discarded.
- DSSP [14] (see below), should provide a valid output file for any chain retained in the dataset.

We started with a dataset containing a total of 4017 protein chains, of which 2656 corresponded to structures determined by X-ray crystallography. Following the above procedure we ended up with 612 protein chains.

Secondary Structure Assignment We have relied for secondary structure (SS) assignments on the widely used DSSP program [14]. More specifically, we reduce class assignments from the eight secondary structure (SS) types provided by DSSP (i.e. α-helix (H) , 3_{10}-helix (G) , π-helix (I) , β-strand (E) , β-bridge (B) , β-turn (T), bend (S) and 'other' ('.')) into three SS states (Helical: H, G; Extended: E, B; Random coil/Loop: I, T, S, '.'). From here onwards, we refer to these states as H, E and C respectively. DSSP results were fetched from the DSSP website (URL: http://swift.cmbi.kun.nl/gv/dssp/, accessed 20 April 2009) and transformed to the 3-state representation by an in-house parser. Since several protein chains contain segments of disordered regions where DSSP does not produce any output, for the purposes of this work we have decided to exclude any such entries.

Multiple Sequence Alignment Preprocessing MSAs have been shown to significantly increase protein secondary structure prediction accuracy in recent applications [8]. This is because structure is considered to be more conserved than sequence [8]. Every position within an alignment contains an evolutionary record. We encode each input residue with a 20-dimensional vector, where ordinates correspond to the frequencies of the different 20 amino acid residues at the respective column of the MSA. Apparently, encoding for single sequences at the input reduces to the orthogonal encoding scheme. For the polypeptide chains collected with the previously described procedure, we have utilised unweighted profiles available from the HSSP database [15].

2.2 Modified BRNN architecture and training procedure

A NN must accept the amino acid at its input with all the necessary information that it needs in order to produce the right output. Taking into account that the formation of different secondary structural elements depends on the interaction between neighboring-in-space (not necessarily in sequence) amino acid residues, we chose the BRNN architecture for its ability to encapsulate information included in the amino acid residues that are coming before and after the residue at the examined position t; where t denotes the discrete time index in $[1, T]$, with T being the total length of the protein chain.

The BRNN architecture consists of two Recurrent Neural Networks (RNNs) and a Feed Forward Neural Network (FFNN). The RNNs are used for taking into account the information contained in a local segment of length L_s centered around position t. The Forward Recurrent Neural Network (FRNN) processes $j = \frac{L_s - 1}{2}$ amino acid residues located on the left side of the segment, computing iteratively from the far left side of the window (i.e., in position $t - j$) and moving towards the right until position t (inclusive) by taking into account a sliding

sub-segment of length S_l. The Backward Recurrent Neural Network (BWRNN) processes the amino acids located on the right side of t, in a similar symmetric way.

During the recurrent network processing, a kind of memory is being formed since the NN correlates each sequence separately and holds an internal temporary knowledge [16]. The output from the two recurrent NNs and the output from the FFNN is correlated and predicts the secondary structure state for residue t as indicated in equation 1.

$$O_t = \eta(F_t, B_t, I_t) \tag{1}$$

where F_t is the forward (upstream) context, B_t is the backward (downstream) context and I_t is the input vector at time (sequence position) t. In the current work, we use an input vector encoding a single residue (corresponding to a window size of unity in Baldi's [1] implementation).

The contextual information from the protein is gathered into a pair of vectors F_t and B_t. Only after the F_t and B_t are computed, the algorithm can predict the state (as in [1]). In order for the amino acids to be examined, two learnable non-linear state transition functions $\phi(\cdot)$ and $\beta(\cdot)$ are applied (equations 2 and 3):

$$F_t = \phi\left(\sum_{j}^{0}\sum_{i=0}^{S_l-1}\gamma F_{t-j+i}, I_t\right) \tag{2}$$

$$B_t = \beta\left(\sum_{j}^{0}\sum_{i=0}^{S_l-1}\gamma^{-1} B_{t+j-i}, I_t\right) \tag{3}$$

where $\gamma \in (0, 1]$ is a modified shift operator, which in effect adds a constant weight based on the importance given to the outputs of the FRNN and BWRNN. Intuitively, we chose $\gamma < 1$ (thus $\gamma^{-1} > 1$) to reflect the fact that protein chains are synthesised from the N-terminal to the C-terminal (i.e., from the left to the right side of the sequence respectively) with some secondary structural elements forming co-translationally [17].

Once the data located within the input vector enters the BRNN, the Mean Square Error function is applied and is used by the Backpropagation algorithm [18] for the BRNN to be trained. The training is performed based on two alternative output encoding schemes: (i) an orthogonal, and (ii) a 'winner-take-all' (WTA). The former scheme has three output units with binary values giving eight possible combinations, three of which are assigned to the three reduced SS states, and the rest are arbitrarily considered to be classified as random coil. The WTA encoding scheme, has three output units as well (corresponding to the reduced SS states) and assigns the SS state of the winning neuron as the prediction for the examined residue. For both schemes, once the error is calculated, the delta rule is applied to update the network weights.

3 Results and Discussion

A set of optimal NN parameters were empirically found following experimentation. For training the network, the dataset of 612 polypeptide chains was randomly split to 513 proteins for training and 99 proteins for testing.

Firstly, we explored BRNN architectures with two hidden layers. More specifically, the FFNN was composed of two fully interconnected hidden layers with 12 neurons each, whereas the BWRNN and FRNN consist of two hidden layers each, with the first hidden layer having 13 and the second 12 neurons.

Our first experiments, utilising single sequences at the input, achieved a highest result of 66.59% (Q3) with optimal parameters: $\gamma = 0.7$, learning rate $\alpha = 0.8$, momentum $m = 0.0$, $L_s = 15$, $S_l = 3$, and orthogonal output encoding.

The next set of experiments was performed using the MSA profiles as input. For 11 out of the 612 protein chains of the initial data set an HSSP profile was not available, thus the data set was slightly reduced to a total of 601 chains, which was again randomly split into a training and a test set of 504 and 97 protein chains respectively. Initially, the predictive accuracy value reached on average 70.82% (Q3, with the aforementioned optimal parameters), almost 4% higher than when training with single sequences. In order to improve the performance, we decided to apply a randomisation procedure on the data at every iteration (i.e., when all the training data is passed through the network). We consider randomisation to be equivalent to the insertion of noise during training, which could theoretically improve the results. As predicted, randomisation does indeed improve the prediction accuracy, as illustrated in Table 1. It has to also be noted that large L_s values gave better results than smaller ones. We experimented with L_s values up to 60 (where data sets were modified accordingly to exclude sequences shorter than L_s), and from the results it was obvious that the optimal L_s was 31 (which concurs with the input window size of [1]). As it can be seen from Table 1 the best result is 73.92% (Q3) accuracy; the corresponding SOV measure is 63.02%. Experimentation with varying the number of neurons in all the layers of the constituent networks of the BRNN did not give any significant improvement in the results reported above.

Table 1. Prediction accuracy results with a BRNN architecture with two hidden layers for different sizes of the local context window L_s. MSA profiles were used as input (for all parameter values, see text).

L_s	15	17	19	21	23	29	31
Q3%	72.94	73.14	72.79	71.80	73.09	73.41	73.92

Secondly, we decided to reduce the complexity of the BRNN architecture by using a single hidden layer. We also changed the output encoding to WTA. In addition, we experimented with different hidden layer sizes and the optimal

results (shown in Table 2, columns 1 and 2) were obtained with: (i) a FFNN of 15 hidden neurons combined with RNNs of 17 hidden neurons (rows 1-4), and (ii) a FFNN of 51 hidden neurons combined with RNNs of 41 hidden neurons (rows 5-8). In order to assess whether these predictions are significantly different, we performed four repetitions of each configuration (see table 2), and descriptive statistics were calculated (data not shown). For the Q3s resulting for the two configurations no statistically significant difference could be observed, with a non-parametric Mann-Whitney test (p-value=0.57). As we can see from the results shown in table 2 there was a significant improvement of the Q3 prediction accuracy measure (up to 76.07%) and a SOV of up to 65.36% compared to architectures with two hidden layers.

Table 2. Prediction accuracy results with a BRNN architecture with one hidden layer, randomized input, WTA output encoding and MSA profiles as input. The first four rows correspond to a BRNN architecture with a FFNN of 15 hidden neurons and RNNs of 17 hidden neurons and the last four rows to a BRNN architecture with a FFNN of 51 hidden neurons and RNNs of 41 hidden neurons. The rightmost columns correspond to the performance metrics after filtering with an HMM and a feed forward ANN.

Q3%	SOV	Q3% HMM	SOV HMM	Q3% ANN	SOV ANN
76.07	64.32	76.57	70.32	76.60	71.90
75.32	64.66	75.17	67.67	75.47	72.90
75.14	62.12	76.38	68.13	75.59	63.05
74.81	65.36	74.69	67.99	75.11	69.88
75.26	64.53	75.75	69.55	76.04	70.40
75.49	64.45	75.90	69.51	76.33	72.91
76.07	62.43	76.84	69.15	76.44	71.19
75.26	65.21	75.90	69.09	73.61	71.32

Knowing that filtering the initial sequence-to-structure network outputs improves the prediction accuracy (see [11] and references therein), we decided to explore this possibility on our BRNN architectures which gave the best results (i.e., with one hidden layer). Two filtering approaches were employed, one based on a novel Hidden Markov Model (HMM) and one based on a cascaded Artificial NN (ANN, similar to the structure-to-structure network of [7]) for comparison. According to the results (shown in table 2, columns 3-6), both filtering approaches improve the results, in particular there is a significant increase in the SOV values (in the order of up to 7%). More specifically, for both configurations the HMM filtering was producing significantly higher SOV values (Mann-Whitney test: p-value = 0.03 for both cases) compared to the unfiltered

results. In addition, the ANN filtering produced significantly higher SOV values only for the second configuration (Mann-Whitney test: p-value =0.03) but not for the first one (p-value=0.2). The increase in SOV was expected, since both filtering procedures are known to eliminate biologically irrelevant predictions, especially in cases where isolated residues are predicted in a SS state.

For our best achieved results, further analysis regarding the details of the prediction in different SS states has been conducted. More specifically, we counted all possible instances of observed versus predicted outcomes, through which confusion matrices were created, as shown in Table 3.

Table 3. Confusion matrices, showing the distribution of predictions for the three secondary structure states, for the best performing configurations: (a) for the best configuration listed in 1 giving a Q3 of 73.92%, (b) best results of the unfiltered configuration shown in table 2 giving a Q3 of 76.07%, (c) best results of the configuration shown in table 2 filtered with a HMM giving a Q3 of 76.57%, and (d) best results of the configuration shown in table 2 filtered with an ANN giving a Q3 of 76.60%. Note that the displayed values correspond to the fraction of predicted residues with a given observed state, expressed as a percentage.

Obs *vs* Pred	H	E	C	H	E	C
		(a)			(b)	
H	62.63	3.39	33.98	71.65	5.46	22.89
E	7.22	45.03	47.74	7.50	59.85	32.65
C	6.55	4.31	89.12	8.01	7.20	84.79
		(c)			(d)	
H	73.24	4.39	22.37	72.74	4.16	23.37
E	5.77	60.62	33.61	0.00	57.99	42.01
C	8.75	7.24	84.01	0.00	14.21	85.79

Three are the main observations from the confusion matrices of Table 3:

1. Helices (H) and loops (C) are pretty accurately predicted.
2. Extended structures (E) suffer from under-prediction. However, the novel HMM-based filtering method seems to partially overcome this problem.
3. Most of the incorrect predictions involve the loop secondary structure state.

Observation 1 was expected since (i) most other works on PSSP report similar trends, and (ii) H and C are the most populated states in our datasets as opposed to strands (observation 2). For observation 3, we believe this could be an artifact of our output encoding scheme (see Section 2.2). It is worth pointing out that the ANN filtering method (see Table 3d) completely eliminates false predictions in the 'H' class.

4 Conclusions

In an attempt to tackle the PSSP problem, our modified training procedure for the BRNN, where the main modification is the updating of the weights for each residue, gives a prediction accuracy of Q3=76.07% with respective SOV = 64.32, which is better than the state-of-the-art results of [1] (Q3=73.6%) and clearly comparable to the results obtained in [1] when an ensemble of 6 BRNN-based predictors was used (i.e., Q3=75.1% on average). Even though we have a slight computational overhead with our approach, where weight updates occur for each residue presented to the BRNN, in practice our BRNNs train (in the worst case) within a couple of hours, and certainly we assume that (for most network parameter sets) our single BRNN would be computationally cheaper than the ensemble of 6 BRNNs reported in [1]. Training time is not an issue for this type of applications, since it not anticipated that the BRNNs will have to be trained very frequently; the most important issue in this application is the prediction accuracy. Certainly, an exact comparison with [1] cannot be made, unless the same training and testing sets are used, but even with that the optimal connectivity of Baldi's architecture is not available (personal communication with one of the authors of [1], G. Pollastri).

In the work of Baldi et al. [1], the BRNN has a short input window of residues centered at the prediction site. Although this small window aims to avoid overfitting, it does not capture variable long-range information, which is overcome by unfolding the RNNs throughout the sequence. Despite the fact that we minimise the input vector in order to contain information for a single residue, the novelty of our approach lies firstly on the fact that we use a more elaborate computation within the recurrent context windows (as described in section 2.2), and secondly on updating network weights at every amino acid residue. With the latter, even though we are not able to capture long-range dependencies, we manage to more accurately take into account all available local information and this seems to be justified from our results.

Improved prediction results were obtained when sequence-to-structure output was filtered, in a post-processing step, in order to take higher order SS correlations into account. In particular, our novel HMM-based filtering approach not only improved the unfiltered results, but it was shown to be on average marginally better than a standard feed forward ANN-based filtering approach and much better than the BRNN-based filtering results reported in [11].

We believe that if we were to use a ensemble of BRNN-based predictors with our training scheme and our novel filtering procedures, our results would be even better.

Acknowledgments

We gratefully acknowledge the support of the Cyprus Research Promotion Foundation, as well as the European Structural Funds for grant TPE/ORIZO/0308 (FR)/05.

References

1. Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G.: Exploiting the past and the future in protein secondary structure prediction. Bioinformatics **15** (1999) 937–946
2. Anfinsen, C.: Studies on the principles that govern the folding of protein chains. Les Prix Nobel en 1972. **1** (1973) 103–119
3. Richards, F., Kundrot, C.: Identification of structural motifs from proteins coordinate data: Secondary structure and first-level supersecondary structure. Proteins **3** (1988) 71–84
4. Pollastri, G., Przybylski, D., Rost, B., Baldi, P.: Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. Proteins **47** (2002) 228–235
5. Rost, B., Eyrich, V.: EVA: large-scale analysis of secondary structure prediction. Proteins **5** (2001) 192–199
6. Qian, N., Sejnowski, T.: Predicting the secondary structure of globular proteins using neural network models. Journal of Molecular Biology **202** (1988) 865–884
7. Rost, B., Sander, C.: Improved prediction of protein secondary structure by use of sequence profiles and neural networks. Proc Natl Acad Sci U S A **90**(16) (1993) 7558–62
8. Rost, B., Sander, C.: Combining evolutionary information and neural networks to predict protein secondary structure. Proteins **19** (1994) 55–72
9. Salamov, A., Soloveyev, V.: Protein secondary structure prediction using local alignments. Journal of Molecular Biology **268** (1997) 31–36
10. Cuff, J., Barton, G.: Evaluation and improvement of multiple sequence for protein secondary structure prediction. Proteins **34** (1999) 508–519
11. Chen, J., Chaudhari, N.S.: Cascaded bidirectional recurrent neural networks for protein secondary structure prediction. IEEE/ACM Trans. Comput. Biology Bioinform. **4**(4) (2007) 572–582
12. Zemla, A., Venclovas, C., Fidelis, K., Rost, B.: A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment. Proteins **34**(2) (1999) 220–223
13. Uwe, H., Michael, S., Reinhard, S., Chris, S.: Selection of representative protein data sets. Protein Science **1** (1992) 409–417
14. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers **22** (1983) 2577–2637
15. Schneider, R., Sander, C.: The HSSP database of protein structure-sequence alignments. Nucleic Acids Research **24** (1996) 201–205
16. Elman, J.: Finding structure in time. Cognitive Science **14** (1990) 179–211
17. Baram, D., Yonath, A.: From peptide-bond formation to cotranslational folding: dynamic, regulatory and evolutionary aspects. FEBS Lett **579**(4) (2005) 948–954
18. Rumelhart, D., Hinton, G., Williams, R.: Learning representations by back-propagating errors. Nature **323** (1986) 533–536