Deep Learning Approach for Predicting Protein Secondary Structure

Punam Biswas[1] **and Taufiq Islam Protick**[2]

[1]**0417052051, 0417052051@grad.cse.buet.ac.bd, Department of Computer Science and Engineering, BUET**
[2]**0417052054, protickrockpit@gmail.com, Department of Computer Science and Engineering, BUET**

Abstract Protein secondary structure prediction (PSSP) is a very significant task in computational biology and one of the fundamental tasks in protein science. It can be utilized to understand other higher level structures and biological functionality of protein. The advent of deep learning in machine learning paved the way to predict protein secondary structure with very good accuracy. To learn about the latest advancements of PSSP through deep learning, this paper provides a brief overview about the progress in this arena. It presents the basic concepts behind the use of deep learning in PSSP. It also assesses the input data features used in deep learning and prediction accuracy of this technique. Finally, it proposes a deep learning based ensemble method whose weighted version can improve the accuracy of individual deep learning based predictor.

Keywords
Protein Secondary Structure Prediction (PSSP), Position Specific Scoring Matrix (PSSM), Deep Neural Network, Ensemble Method in PSSP, Weighted and unweighted voting.

1 Introduction

Protein is a very important molecule and the material basis of any living cell as it guides several biochemical process and gene expression. Advanced technology lets us collect protein sequence data correctly. The functionality of protein largely depends on its structure which can be categorized in some hierarchical level: such as primary, secondary, tertiary and quaternary. Secondary structure of protein works as the bridge between primary and tertiary structure of protein and it is the basis for the spatial structure of a protein.

To understand the spatial structure and as such the functionality of protein, correctly identifying or predicting protein secondary structure is indispensable. The actual protein secondary structure can be determined with high precision by different mechanical methods like X-ray crystallography and multi-dimensional magnetic resonance etc. But, these methods are very costly and time consuming and so not feasible for finding secondary structure of huge amount of protein. That's why only 0.2% of the ever-increasing dataset of proteins have a known structure. Here comes the need for finding computational methods that can predict with minimum error the secondary structure of protein. Computational methods provide simple, low cost and most importantly fast speed techniques for prediction. This problem is referred to as Protein Secondary Structure Prediction (PSSP) in computational biology and bioinformatics.

Secondary structure prediction accuracy largely depends on the definition style of secondary structure. The most widely used standard is the secondary structure assignment method Dictionary of Secondary Structure of Proteins (DSSP) [1]. This standard automatically assigns secondary structure into eight states based on hydrogen-bonding patterns. These eight states are commonly simplified into only three states namely helix, sheet and coil. The most widely used convention is that helix is represented as G (3_{10} helix), H (α-helix) and I (π-helix); sheet as B (isolated bridge) and E (extended sheet); and all other states designated as coils.

Figure 1 shows the three major states of protein secondary structure.

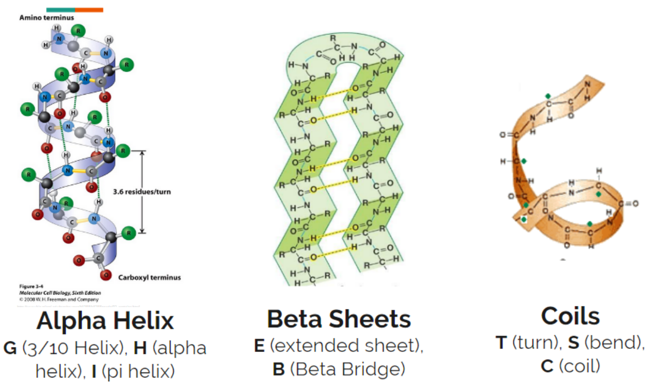
Amino terminus
3.6 residues/turn
Carboxyl terminus
Alpha Helix
G (3/10 Helix), H (alpha helix), I (pi helix)
Beta Sheets
E (extended sheet), B (Beta Bridge)
Coils
T (turn), S (bend), C (coil)

Figure 1. Secondary Structures of Protein

Scholars are in search of powerful techniques for PSSP since helical and sheet conformations for protein polypeptide backbone was predicted in 1951. First generation PSSP (before the 80s) [2, 3, 4] used statistical probability method on individual residue information of protein. Chou–Fasman method [4] is the most representative first-generation method. The second generation methods (from 1980 to 1992) leveraged neighboring residues information and physico-chemical information. They used a sliding window and several efficient algorithms from statistics and machine learning. Representative methods of second generation are Garnier-Osguthorpe-Robson (GOR) method [5] and the Lim method [6]. Third generation PSSP methods (from 1992 to 2006) utilized multiple sequence alignment profile, homogeneous information and long range correlation of residues. These methods mainly used different machine learning algorithms (support vector machines [7, 8], Bayesian or hidden semi-Markov network [9, 10], conditional random fields etc. [11]) as the backbone architecture. Current methods of PSSP mainly uses advanced machine learning techniques and hybrid models. These methods incorporate multiple sequence alignment profile, homogeneous information, long-range correlation and other protein natural properties.

Artificial neural network (ANN) [12] is a major breakthrough in machine learning. A deep learning (DL) method [1]uses an ANN of multiple hidden layers to find the complex relationship between various input features. Applications of deep learning has got tremendous success in problems of diverse research fields. This success urged researchers in bioinformatics to utilize deep learning method in PSSP.

Figure 2 shows a schematic diagram of deep learning architecture.

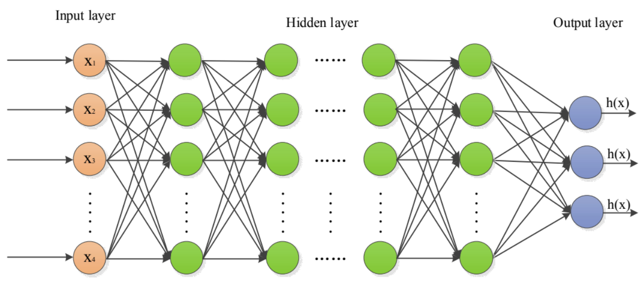

Figure 2. A schematic diagram of deep learning architecture

Deep learning utilizes multiple hidden layers to get more abstract representation in the higher levels, which is the combination of the features in lower levels. By this way, this model can find more advanced features from raw data which can be incorporated usefully to predict the final output. The pattern recognition of PSSP is highly nonlinear and complex problem. Moreover, the distribution and combination of protein sequence features is

[1]https://en.wikipedia.org/wiki/Deep_learning

high dimensional and show much diversity. Therefore, researchers try to explore deep learning method in PSSP due to its huge prospect in high dimensional and variable bio-data processing.
We explore existing DL techniques for PSSP and find that DL techniques have great potential to reach the theoretical highest limit (88–90%) [13] of PSSP prediction accuracy. We built a small scale deep learning network and utilize PSSM to verify effectiveness of using this type of input in deep learning method for PSSP. We also propose an ensemble method that uses deep learning based individual predictors. Experiments on this method shows that, our weighted voting technique can gain better accuracy than independent predictors used in this ensemble method. We took the best known DL based predictor [14, 15, 16] as one of the predictors in our ensemble method to leverage current best DL method for PSSP in combined prediction.

The major contributions of this paper are as follows:

- We provide a comparative overview on the existing deep learning based PSSP techniques.
- We verify the performance of using a common input feature (PSSM) in DL method.
- We propose a weighted voting process for PSSP which primarily have deep learning based predictors.
- Finally, we designed a neural network of modifiable hidden layers and other parameters and tested with a small set of proteins to evaluate what combinations of parameters provided the best accuracy.

2 Related work

In the following subsections, we give short overview of the existing deep learning based approaches for PSSP. We categorized the related works based on the underlying artificial neural network which is at the core to deep learning approach.

2.1 Feed Forward Neural Network Based

In 2014, Spencer et al. proposed the first deep learning based PSSP method, called DNSS, and it was a deep belief network (DBN) model based on restricted Boltzmann machine (RBM) and trained by contrastive divergence in an unsupervised manner [17]. The method used PSSM generated by PSI-BLAST to train the deep learning based network. Due to the difficulty in training the network with complex calculation in different layers and for the requirement of a large number of training samples, the authors applied graphical processing units (GPU) and CUDA software to optimize the model. Their work confirmed that deep learning could contribute to the progress in the field of PSSP.
Heffernan et al. showed that more additional features can further improve the prediction accuracy of PSSP; therefore, they proposed an iterative features based PSSP method including solvent accessible surface area, backbone angles and dihedrals based on Carbon alpha atoms. By considering several features of different physiochemical properties of amino acid and PSSM residues, they designed a deep learning based PSSP method which showed that the evolutionary information of protein would further improve the prediction accuracy of secondary protein structure [18].
In 2016, Wang et al. proposed a deep convolutional neural fields (DeepCNF) model which was mainly a deep learning extension of conditional neural fields (CNF) for PSSP. This method includes 3-state and 8-state PSSP. The DeepCNF combined the advantages of both deep convolutional neural networks (DCNN) and CNF. It also considered the complex sequence-structure relationship and inter dependency between adjacent secondary structure labels, and longer-range dependencies information [14].
As an add-on to this research work, Wang et al. further proposed a deep recurrent encoder-decoder networks for PSSP, named secondary structure recurrent encoder-decoder networks (SSREDNs) [16] . SSREDNs integrated deep architecture and recurrent structures to mapping relationship between input protein features and secondary structures. It also incorporated mutual interaction among continuous residues of protein in the input layer.
Table 1 is a summarized view of feed forward neural network based deep learning approaches for PSSP.

Table 1. A brief summary of feed forward neural network based deep learning

Dataset	Accuracy (%)	Year	Reference
CASP (CASP9, 105 proteins) CASP10, 93 Proteins	Q3=80.7 SOV 74.2	2015	[17]
CASP11	Q3=81.8	2015	[18]
CASP;CB513	Q3=84.7, SOV=86.5 Q8=72.3; Q3=82.3, SOV=84.8	2016	[14]
CB513; CullPDB	Q3=82.9, Q8= 68.20	2017	[16]

2.2 Recurrent Neural Network Based

A recurrent neural network (RNN) is a class of artificial neural network where connections between nodes form a directed graph along a sequence. This allows it to exhibit temporal dynamic behavior for a time sequence. Unlike feed forward neural networks, RNNs can use their internal state (memory) to process sequences of inputs [2]. In an RNN, the information cycles through a loop. While decision making, it considers the current input and also what it has learned from the inputs it received previously. In short, RNNs add the immediate past to the present[3].

[2]https://en.wikipedia.org/wiki/Recurrent_neural_network
[3]https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5

Figure 3 shows a schematic diagram of basic RNN.

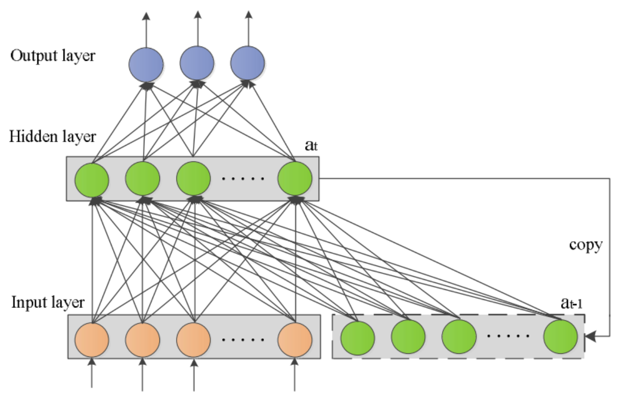

Figure 3. The schematic diagram of RNN (Figure from Jiang et. al [19])

The unique algorithm for state processing in RNN makes it a very good candidate for background neural network of deep learning. Researchers [20, 21, 22, 23, 24] used RNN based deep learning for PSSP to improve hidden layer data processing.
Basic RNN has a limitation of not reaching future information from current state. Hence, Pierre et al. proposed bidirectional recurrent neural network (BRNN) into PSSP. Their approach basically connects two non-causal hidden layers of opposite directions to incorporate both upstream and downstream information [25]. This method overcome the aforementioned shortcoming of basic RNN and showed better prediction accuracy in PSSP. Their success urged researchers to modify basic RNN model and implement it for PSSP to get better result.
Kountouris et al. introduced different prediction methods based on an ensemble of six BRNN models and filtering methods architectures. Chen et al. established cascaded BRNN which utilized the outcome of the first BRNN (sequence-to- structure BRNN) as the input of second BRNN (structure-to-structure BRNN) to get final prediction [21]. Mirabello et al. proposed yet another cascaded BRNN based PSSP method in 2013. The first BRNN took primary sequence and multiple sequence alignments(MSA) as input and predicted secondary structure, whereas the second BRNN filtered the predictions of the first BRNN [23]. This type of cascaded architectures leverages nonlocal information and secondary structure-structure correlations of protein.
Babaei et al. proposed a modular prediction system for PSSP that integrated a multilayer bidirectional recurrent neural network (MBR-NN). This architecture utilized the local co-relations of secondary structure elements. It used the multilayer reciprocal recurrent neural networks (MRR-NN) as backbone method to capture the long-range intramolecular interactions of amino acids [22].
Human memory system inspired J. Chen et al. to introduce a segmented-memory recurrent neural network (SMRNN). This method breaks the entire input sequence to segments and memorizes each segment. Finally, the segments are cascaded to get the final sequence. The main contribution of SMRNN based PSSP method is to capture long-term dependencies in proteins with tremendous efficiency [20]. Recently Heffernan et al. proposed a long short term memory BRNN (LSTM-BRNN) method for PSSP that would avoid the shortcomings of sliding window based methods. It successfully extracts the long-range features in proteins [24]. Both of these modified RNN based methods tried to simulate human memory information propagation and capture long-term dependencies of protein.
Table 2 summarizes RNN based deep learning approaches for PSSP.

Table 2. A brief summary of RNN based deep learning

Dataset	Accuracy (%)	Year	Reference
CB396 (RS126 set for training)	Q3=73.1 SOV= 63.0	2006	[20]
PSIPRED (training dataset :EVA)	Q3=74.38 SOV= 66.05	2007	[21]
PSIPRED	Q3 = 79.36 SOV= 70.09	2010	[22]
1630 proteins of lower quality from PDB	Q3=82.2	2013	[23]
TS115	Q3=83.9	2017	[24]

2.3 Radial Basis Function Neural Network Based

Radial basis function neural network (RBFNN) is a special kind of neural network that uses radial basis function[4] as its activation function of hidden layer, where Gaussian function is used typically. It is a non-linear multilayer feed forward neural network that can find the best fitting plane in hidden layer (high dimensional plane) by the mapping relationship of input layer (low dimensional space). Based on function approximation theory [5], RBFNN can approximate any continuous function with arbitrary precision. Hence it is suitable for solving the non-linear classification problem of PSSP.
While studying 340 protein sequences and their corresponding secondary structures from PDB, Zhang et al. classified 20 amino acids into three groups, as f (Former), b (Breaker) and n (Neutral). This classification information reflects the inherent characteristics of different residues. By utilizing this information, the input vectors of RBFNN can be reconstructed to get better accuracy of PSSP [26]. The main intuition behind using this technique is to improve input vector information to gain better prediction.
Researchers [27, 28] modified existing traditional RBFNN architecture for PSSP. Jing et al. [27] introduced a two-level RBFNN structure by the model of sequence-to-structure-to-structure that used the output of first layer

[4]https://en.wikipedia.org/wiki/Radial_basis_function
[5]https://en.wikipedia.org/wiki/Approximation_theory

as the input of second layer and also considered the evolutionary information of proteins.
The RBF neural network can overcome many limitations of traditional feed forward neural network (such as BPNN). It can avoid local minimum problem due to its only best approximation characteristics. RBFNN is superior to BPNN on the characteristics of good classification ability and approximation performance. And its input-output mapping function and learning speed is much better than other feed forward neural networks. But the main challenge of RBFNN is to find the center of the hidden nodes which the key to improve approximation.
Table 3 shows a brief summary of RBFNN based deep learning approaches for PSSP.

Table 3. A brief summary of RBFNN based deep learning

Dataset	**Accuracy (%)**	**Year**	**Reference**
340 protein from PDB	Q3=77.4	2005	[26]
RS126	Q3=71.3	2008	[28]

3 Our Method

In this section, we will describe two methods that we used to predict the secondary structure of proteins. Firstly, we wanted to predict the secondary structure of proteins by taking the weighted voting of some online server tools that have high accuracy, uses neural network and uses recent approaches towards the problem. Secondly, we built a neural network on our own that uses the Position Specific Scoring Matrices (PSSMs) as input features generated by RaptorX-Property, one of the online server tools. Section 3.1 discusses about the weighted and unweighted voting methods and Section 3.2 discusses the findings of our approach towards building a neural network.

3.1 Combined Voting Method

A brief description of the server tools, followed by the weighted and unweighted voting techniques are described in this section.

3.1.1 Server Tools used

We used four Server Tools in our combined voting system. All these server tools are termed as predictors in the subsequent part of the Section. Their brief description is given below:

1. RaptorX-Property[6]: The server is described in[15]. It is a web server for predicting structure property of a protein sequence without using any templates. It outperforms other servers, especially for proteins without close homologs in PDB or with very sparse sequence profile (i.e. carries little evolutionary information). To predict structure properties, this server employs a new machine learning model DeepCNF (Deep Convolutional Neural Fields) [14].

2. SPIDER2[7]: This is a web server tool that uses iterative deep learning to predict the secondary structure of the proteins, torsion angles and ASA (Accessible Surface Area). The paper [18] describes the methodology implemented in this server tool. It has an 82% accuracy for secondary structure prediction.

3. Porter 4.0[8]: This server uses the methodologies of this paper[23]. The paper presents an ab-initio prediction of protein secondary structure and solvent accessibility. Porter 4.0 predicts secondary structure correctly for 82.2% of residues. It is based on ensembles of BRNN (Bidirectional Recurrent Neural Network).

4. Jpred4[9]: The server is described in [29]. It incorporates Jnet[30]. In this paper. The network consists of two ANNs.
   A brief summary of the types of neural networks used in the servers and their accuracy can be found in the Table 4.

[6]http://raptorx2.uchicago.edu/StructurePropertyPred/predict/

Table 4. A brief summary of the Server tools used in combined voting

Server Tools	Methods Used	Q_3(%)
RaptorX-Property	Deep Convolutional Neural Fields (DeepCNF)	82.3
SPIDER2	Iterative Deep Learning	81.9
Porter 4.0	Bidirectional Recurrent Neural Network (BRNN)	82.0
Jpred4	2 Artificial Neural Networks	77.1

3.1.2 Unweighted Voting Method

The unweighted voting method gave a weight of $1/p$, where p is the number of predictors, to all the predictors and assigned their votes to a particular amino acid's secondary structure prediction. As we used four predictors, each had 1/4 or 25% weights in their vote in this method. The secondary structure that got maximum score (i.e. maximum weight) combined was assigned as the target structure for an amino acid. Any case of tie was resolved by randomly choosing one secondary structure over another with equal probability.

3.1.3 Weighted Voting Method

This method is divided into two stages. In the first stage, some proteins were used to find out the predictors' Q_3 accuracy. Next, weights were assigned to them based on their accuracy by normalizing their accuracy rates, i.e., a

[7]http://sparks-lab.org/server/SPIDER2/
[8]http://distillf.ucd.ie/porterpaleale/
[9]http://www.compbio.dundee.ac.uk/jpred4/index.html

predictor with higher Q_3 accuracy got higher weights in voting. Like the unweighted method, the weights here sum to 1.
In the second stage, these four predictors with their assigned weights were used on a new set of proteins to predict their secondary structures and were checked to find out whether they are performing better than any of the individual predictors alone. Any case of tie was resolved by randomly choosing one secondary structure over another with equal probability.

3.2 Description of our Neural Network

The Neural Network has been designed from scratch, in Java. It is a feed-forward neural network that learns with back propagation.

3.2.1 Dataset Used

We used 50 proteins from the DSSP (Dictionary of Secondary Structures of Proteins) [1] data set that holds the ground truth for secondary structures of roughly 370,000 proteins. We used 45 proteins for training and 5 for testing the neural network.

3.2.2 Input Layer

The input layer passes the PSSM profile of a protein window-wise. The PSSM profile for each protein was obtained from the RaptorX-Property server. The values of the PSSMs were divided by 100 and then rounded. Next, they were either passed as input features unscaled, or scaled to a value in the range [0,1] by the sigmoid function.

3.2.3 Output Layer

The output layer had three perceptrons, each holding the sigmoid value of the weights of its neurons multiplied with the input vector. These three perceptrons represented three secondary structures, α-helix, β-sheet and coils in order. The perceptron with the highest output value among the three was the winner and the secondary structure of the central residue of the window was decided based on the winner perceptron. For example, if for a window=17 of proteins, say "MVLSEGEWQLVLHVWAK" the second perceptron of the output layer had the highest sigmoid output, it was decided that the central amino acid of the sequence "MVLSEGEWQLVLHVWAK", i.e. "Q" is in a β sheet secondary structure.

3.2.4 The learning procedure

As the true output for a window of protein is already known in training data, the weights in the output layers can be learned and corrected in each iteration. But in a multilayer neural network, the hidden layers, unlike the output layer, don't have a fixed agenda, i.e. unlike the output layer, they don't have a fixed output to show. Hence, the errors are propagated backwards, so that the hidden layers next to the output layer can modify its weights by the error of the output layer, and the second to last hidden layer can modify weights from the errors of the last hidden layer, and so on. For this task, the back propagation algorithm described in [12] was used. A slight review of the back propagation algorithm is needed to gain insight of how the network is learning in each iteration.

After the neural network is designed, the first job was to initiate the weights of each perceptrons randomly. In our project, we have initiated the random weights in the range [-0.05, 0.05]. Next, for each training dataset (i.e. for each window of PSSM profile), the input was fed forward through the network window by window. Afterwards the errors are back propagated in the following way.

At first, for each of the output unit k, the error term δ_k is calculated by the following formula.

$$\delta_k \leftarrow o_k(1-o_k)(t_k-o_k) \tag{1}$$

Here o_k is the output of the neural network and t_k is the target output(i.e. what the output was supposed to be if it predicted the structure 100% correctly) of the neural network.

Next, for each hidden unit h, the error term δ_h was calculated by the formula:

$$\delta_h \leftarrow o_h(1-o_h)\sum_{k\epsilon outputs} w_{kh}\delta_k \tag{2}$$

Here, o_h is the output of the hidden layer, w_{kh} is the weight associated with the h^{th} input to unit k.

Finally, the weight was updated in the entire network by the following formula

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji} \tag{3}$$

where,

$$\Delta w_{ji} = \eta\delta_j x_{ji} \tag{4}$$

Here, x_{ji} means the i^{th} input to unit j, η is the learning rate, and w_{ji} has the definition same as the definition above in Equation 2.

4 Results

In this section, we will discuss about the findings of our approaches toward protein secondary structure prediction. In Section 4.1, we will show our findings on combined voting method and in Section 4.2, we will show the findings of the designed neural network.

4.1 Result of the combined Voting of the Server Tools

In this section, we will describe the dataset used, the individual Q_3 accuracy of all the four predictors in this dataset. Next, we will use a new set of proteins, where we will do unweighted voting and weighted voting. After that, we will run all the four predictors individually on them to find out whether the weighted voting performed better than their individual performance.

4.1.1 Dataset used

In this experiment, we took 15 proteins from the DSSP, and passed them as text sequences in the online servers of the four predictors. After a short time, their predictions were complete and the assigned secondary structures of the predictors were saved in separate files. Next, these predictions were compared against the ground truth of the DSSP to find out which predictor had the highest Q_3 accuracy.

4.1.2 Overall Q_3 accuracy of the four server tools

After collecting the predicted secondary structure from four server tools, we compared it with the ground truth and found out that RaptorX-Property is performing the best over the data while JPred is performing with the least accuracy rate. JPred4, Porter 4.0, RaptorX-Property and SPIDER2 had, over the set of 15 proteins, an accuracy of 76.62%, 81.8%, 82.67% and 82.07% respectively.

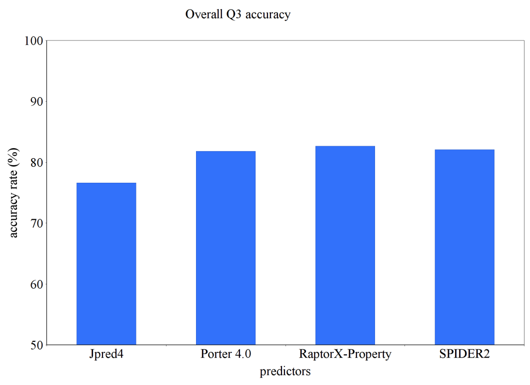

Figure 4. Overall Q_3 accuracy rate over a set of 15 proteins

Although from apparent observation it might seem that accuracy of JPred4 is the least, the protein-wise accuracy of JPred4 is not always the worst one. For example, in the protein with accession id 9PAI:B and 101M:A, the Q3 accuracy of JPred4 was the highest. This calls for further research towards choosing a specific server tool for special proteins.

4.1.3 Overall Q_3 accuracy of unweighted voting

As we are not assigning weights based of the overall Q_3 accuracy on 15 proteins, we have used the same set of 15 proteins here and did likewise. The average Q_3 accuracy of unweighted voting was **82.76%** which is also better than any of the individual prediction accuracy. The weighted accuracy was **83.985%** in that same set of proteins.

4.1.4 Overall Q_3 accuracy of weighted voting

As we cannot use the weights of the predictors to the same dataset from which the weights were calculated, we are using a different set of five proteins on which we will be trying to test whether weighted voting works better than any individual predictor.

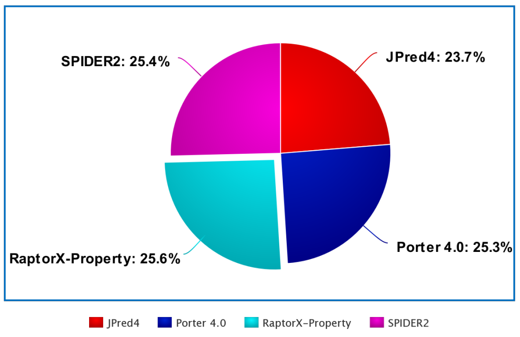

Figure 5. These are the assigned weights to four predictors according to the Q_3 accuracy obtained from 15 proteins. Vote of each had 25% weightage before, that now have changed. The cyan part (piece out of the pie) reflects that RaptorX-Property's vote has the highest weight.

In the previous experiment (the one with 15 Proteins) the weights were normalized to 1 and JPred4, Porter 4.0, RaptorX-Property, and SPIDER2 were assigned weights 0.23709, 0.25314, 0.25582, and 0.25396 respectively (see Figure5). With these weights the average Q_3 accuracy was **87.48%**.

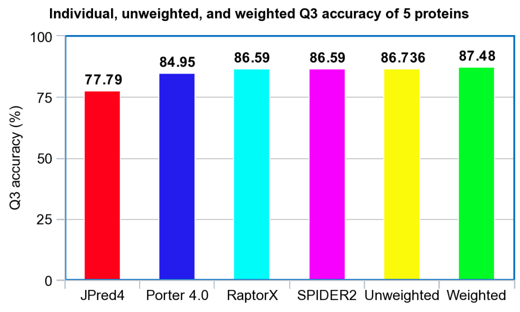

Figure 6. The Q_3 accuracy of 5 proteins used later to evaluate the weighted voting system. The weights calculated on a set of 15 proteins previously was applied to a different set of 5 proteins to predict the secondary structures.

After we were done with the weighted observation, the accuracy was compared against the individual Q_3 accuracy and unweighted Q_3 accuracy of the predictors. It is seen in Figure 6 that, the Q_3 accuracy of the weighted voting is better than individual and unweighted ones.

4.2 Results of the designed Neural Network

In this section, we will discuss the findings of secondary structure prediction by our neural network. In the following sections, we will be discussing about the dataset used, and effect of various factors over the Q_3 accuracy of the neural network.

4.2.1 Some Notations for Convenience

In the following sections, some parameters will be frequently used and their elaborated form is quite redundant. That is why we will be using some notations that would be convenient for the reader.

Table 5. Some notations used throughout the report

Parameter	Explanation
PC	An array that describes the number of perceptrons from input layer to output layer over the entire network. For example {17, 50, 50, 3} means this NN has two hidden layers with 50 neurons each. An input layer with window 17, and an output layer that classifies with 3 neurons.
Sigmoided entry	Whether or not the PSSM entries were scaled in the range [0,1] by sigmoid function
weight range	Initially the neural network is assigned random weights. This range describes the lower and upper limits of these random weights
epochs	# of times the training data passes through the network
W	size of a window that moves one residue per iteration over the entire protein sequence
LR	the learning rate
var	a parameter that is variable

4.2.2 Dataset used

A set of 50 proteins from DSSP was selected as the dataset and 45 of them were used to train the neural network. The BLAST tool shows that pairwise local alignments of all the proteins have sequence similarity in a very small coverage of the entire sequence.So it can be inferred safely that the test dataset of 5 proteins has an overall sequence similarity less than 25% with the sequences of 45 proteins of the training dataset. It is to be mentioned that, all the average accuracy rates shown in Graphs and Bar Charts in the subsequent Sections was an average of five readings run with the same parameters over a neural network.

4.2.3 Effect of Learning Rate over the network

The learning rate, a rate at which the neural network corrects its weights in the back propagation stage, has an immense impact over the Q_3 accuracy over the network. The impact of learning rate is shown in Figure 7 below.
In Figure 7, the bar chart shows the effect of learning rate over the network. It is evident from the figure that the small learning rate contributes less to the accuracy of the

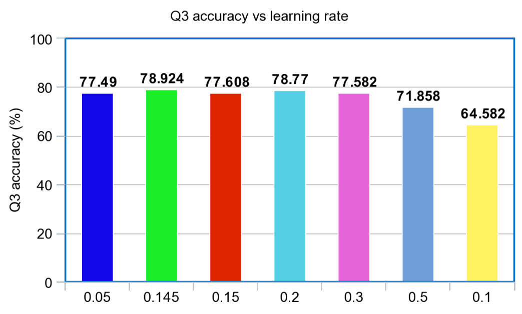

Figure 7. The Q_3 accuracy vs learning rate. The other parameters are PC= {17,50,3}; sigmoided entry= No; weight range [-0.05.0.05]; epochs=20; window=17

Table 6. Summary of Figure 7

Max/Min	Learning Rate	value(%)
Max avg. accuracy	0.145	78.924
Max accuracy of all readings	0.25	80.17
Min avg. accuracy	1.0	64.582
Min accuracy of all readings	1.0	60.2

neural network. As the learning rate increases, the accuracy goes up. But after 0.2 the learning rate gradually goes down, giving the bar chart a hill-shape. After 0.3, there is a remarkable decrease in the accuracy rate, which further goes down when the learning rate reaches 1.0. Table 6 summarizes the maximum and minimum accuracy of the average and all readings.

4.2.4 Effect of Window Size

Fixing the learning rate at 0.145, we varied the window size from 13 to 25 over the neural network. Window sizes were odd values to keep only one central amino acid in the middle. Figure 8 shows the findings. In Figure 8 it is

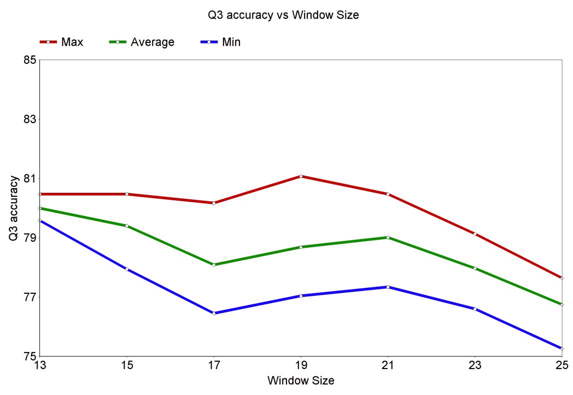

Figure 8. The Q_3 accuracy vs window size. The other parameters are PC = {W,50,3}; sigmoided entry = no ; weight range = [-0.05.0.05]; epochs = 20; LR = 0.145

clear that as the window size increases, the Q_3 accuracy decreases. The maximum average accuracy (79.996%) occurs in the very beginning at W=13. Although the maximum accuracy of all the readings (81.073%) was found at W=19. The minimum average accuracy (76.748%) occurs at W=25.

4.2.5 Relationship between epochs and number of perceptrons in the only hidden layer

In this section we showed the relationship between epochs and number of perceptrons in the only hidden layer. It can be understood by intuition that, any network having more perceptrons in the hidden layer must need more epochs to converge. It was evident for epochs 20 and 30, i.e. increasing the number of perceptron needs increasing number of epochs to converge. In case of 40 epochs the result was noisy. To sum up, it can be inferred that a small number of perceptrons in the hidden layer needs less number of epochs to converge, and vise versa. Figure 9 shows the comparative relationship between epochs and the number of perceptrons in the only hidden layer.

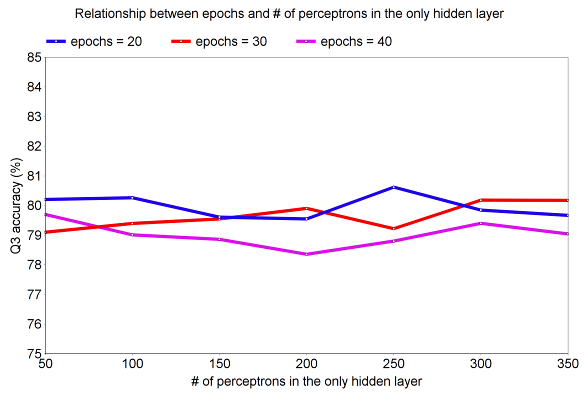

Figure 9. Q_3 **accuracy vs number of perceptrons in the only hidden layer for three different epochs. The other parameters are PC = {13, var, 3}; sigmoided entry = No; weight range= [-0.05.0.05]; epochs = var (20 to 40); LR = 0.145; W = 13; # of perceptrons= var (50-350)**

Table 7. Summary of Figure 9

Max/Min	epochs	# of perceptrons	value(%)
Max avg. accuracy	20	250	80.62
Max accuracy of all readings	20	350	82.11
Min avg. accuracy	40	200	78.356
Min accuracy of all readings	40	250	76.45

In Figure 9, it is seen that, when the number of perceptrons in the only hidden layer is small, a network gives better accuracy with small number of epochs. Although it is not true for the observation of the highest average accuracy in this experiment. In Figure 9, the highest average accuracy (**80.62%**) occurred with epochs = 20, and with 250 perceptrons in the only hidden layer. The red (epochs = 30) and magenta (epochs = 40) crosses their path after 50 perceptrons in the hidden layer and it is seen that epochs = 40 always gave the worst accuracy on the subsequent readings. A comprehensive summary of Figure 9 can be found in Table 7.

4.2.6 Effect of more hidden layers

In this section, we will try to make the neural network deeper by inserting more hidden layers to it. We have increased the depth from two to four and tried to observe how the neural network behave. A comparative study is shown in the bar chart below.

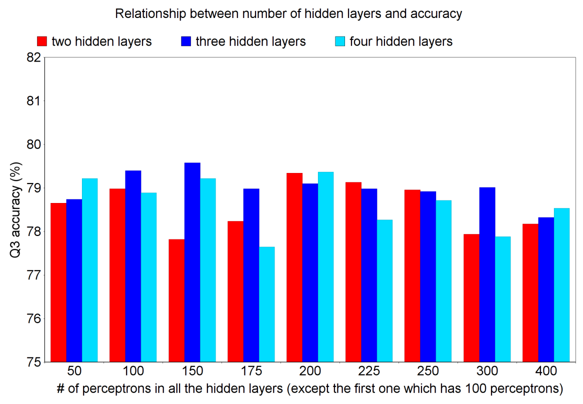

Figure 10. A comparative bar chart showing the effect of more than one hidden layer upon the Q_3 accuracy. The other parameters are PC={13,100, var, var, var, 3}; sigmoided entry=no; weight range=[-0.05.0.05]; epochs=20; window=13; LR=0.145; # of hidden layers= var (from 2 to 4)

All the neural networks in the comparison study had 100 perceptrons in the first hidden layer. The rest were variable from 50 to 400. In Figure 10, the maximum average accuracy was **79.578%**. This was achieved by a neural network with **three hidden layers, and 150 perceptrons in each hidden layer except the first one**. Throughout the entire study, the maximum (not average) accuracy of all readings achieved an accuracy of **81.96%**. This too was achieved by a neural network with **three hidden layers, with 300 perceptrons in each hidden layer except the first one**. The minimum average accuracy (**77.644%**) was achieved by a neural network with four hidden layers, and 175 perceptrons in each hidden layer except the first one.

If we look closely in Figure 10, we will see a fluctuating property of accuracy when four hidden layers are used, i.e., increasing the number of perceptrons in the hidden layers had no particular effect when the neural network had four hidden layers. On the other hand, a neural network with two or three hidden layers had a gradually increasing and then decreasing property, i.e., as the number of perceptrons in the hidden layers increases, the accuracy goes up, becomes maximum somewhere in the middle, and then goes down as the number of perceptrons are further increased. For two hidden layers, it starts at 150, tops at 200, and next goes down. For three hidden layers the same things happens at 50, and the top value is found at 150. To summarize, a neural network with four (or perhaps more) may not have any significant effect on the accuracy on such a dataset.

4.2.7 Sigmoided vs Non sigmoided PSSM Entry

The idea of scaling the PSSM entry with sigmoid function was mentioned in [17]. In their paper, they took the PSSM entries as feature only after scaling them with a sigmoid function. The comparative analysis of the impact of PSSM entry sigmoided vs non sigmoided on accuracy is shown in Figure 11.

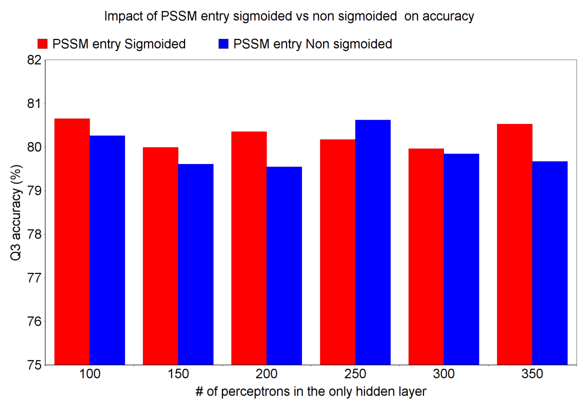

Figure 11. A comparative bar chart showing the effect of PSSM entries sigmoided vs non sigmoided. The other parameters are PC={13,var,3}; sigmoided entry=yes(red) and no (blue); weight range=[-0.05.0.05]; epochs=20; window=13; LR=0.145; # of perceptrons in the only hidden layer= var (from 100 to 350)

It is surprising that, if the PSSM entries are scaled by a sigmoid function, the accuracy is better than the non-sigmoided counterpart. In Figure 11, most of the average accuracy in the sigmoided counterpart is above 80%. Although the maximum average accuracy of the sigmoided counterpart (**80.65%**) does not have a significant difference over the maximum average accuracy (**80.62%**) of the unsigmoided counterpart.

4.2.8 Making the Sigmoid function more or less steep

If we define sigmoid function as $\sigma(x) = \frac{1}{1+e^{-\alpha x}}$, then at $\alpha = 1$, the function is at its standard form. The function becomes steeper if $\alpha > 1$ and the function becomes less steep or softer in the range $0 \le \alpha < 1$. This Section discusses the relation between accuracy and steeper or softer version of the sigmoid function in scaling the PSSM entries. Figure 12 shows the analysis.

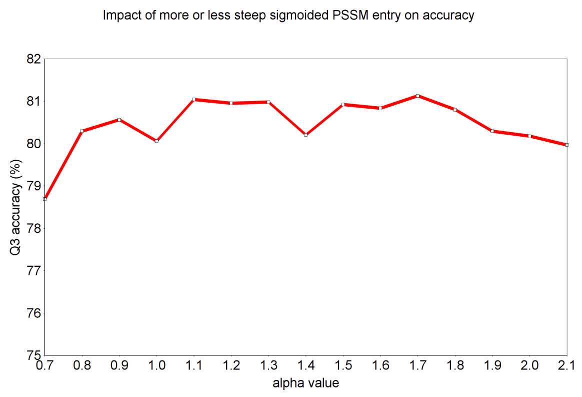

Figure 12. A Graph showing the impact of using steeper or softer sigmoid function in scaling the PSSM entries. The other parameters are PC={13,100,3}; sigmoided entry=yes; weight range=[-0.05.0.05]; epochs=20; window=13; LR=0.145.

In Figure 12, we see that a softer sigmoid function in scaling the PSSM always functions poorly than a standard one. On the other hand, as the steepness rises, i.e. α increases, the accuracy starts to increase than a standard one. However it again starts to go down as α crosses 2.0. It was found that a steeper version where $\alpha > 2.0$ gives less accuracy than when α is between 1.0 and 2.0.

5 Discussion

The success of Protein Secondary Structure Prediction depends largely on the feature selection in input layer. Researchers are continuously searching for new features of protein or amino acids. Incorporating these features in the input layer of deep learning network may prove to be useful for increasing prediction accuracy. Furthermore, the network configuration is an important aspect for correct prediction. Hence, finding the optimal network configuration is another big challenge in PSSP through deep learning. Another direction for future research in PSSP is to check Deep Learning method performance for newer protein data sets.

As future works, we have a lot of tasks ahead. In this project, for the combined voting system, we used the Web version of the predictors and tried to evaluate the voting's performance by manually downloading the data. In case of analyzing large scales of data, such an approach will be very time consuming. Therefore, in future, we will try to use the desktop version of this servers and will try to analyze the combined voting system with large number of proteins to get a general view.

In Section 4.1.2, we observed that, there are some proteins in which a server tool with sub-optimal accuracy is performing better than the server tool with the best accuracy. In those cases, only weighted voting is not enough. A protein-wise accuracy analysis is needed to find out whether the top server tool is performing the best in all the proteins' secondary structure prediction, or there are some cases where the some sub-optimal predictors are performing better. If it is the latter case, then the protein in which the predictor is performing well should be taken under scrutiny, their common properties should be found out. Next, in the final classifications where the predictors are put to test, if proteins of such properties arrive, then the sub-optimal predictor which gave the best accuracy on similar proteins could be given the highest weight and it could be observed whether such a voting system can improve the accuracy or not.

The neural network we built from scratch uses only one feature, i.e., the PSSM as its input. We speculate that is why its accuracy is not yet as high as the state-of-the-art techniques available now. Moreover, it uses the PSSM values obtained from RaptorX-Property, one of the server tools we used in the combined voting prediction method. This has limited our endeavors in putting a large number of protein sequences in our training set. As the procedure was quite manual and due to time constraints, we had to keep the training data limited to 50 proteins. In future, we would try to incorporate more and more features by some deeper research over amino acid. Moreover, we will try to generate PSSMs from PSI BLAST using scripts or other automated way rather than manually downloading it from some other servers. This would help us to analyze our neural networks in a larger scale.

The dataset that we used was very small. In the future, we hope to work with some standard dataset to observe whether or not our designed neural network is performing up to the mark with those datasets as well.

There are a lot of dataset dedicated solely to protein secondary structure prediction. One of them is the CASP dataset, which is updated regularly. As a future work, we will try to incorporate CASP data set in training and testing of our neural network. This way, we could compare our approach with the other state-of-the-art techniques.

A neural network has a lot of parameters, and the combinations of all these parameters are even more. In this project, we could tweak only a few. In most cases, tweaking the parameters of a neural network in an exhaustive manner can produce useful information about the network, i.e. how many hidden layers, how many perceptrons in each hidden layer, which window size, how many epochs or what kind of random weights in the initial stage of the perceptrons will be best suited for prediction. In future, we would want to search more of the combinations of the parameters for better accuracy.

6 Conclusion

Protein Secondary Structure Prediction (PSSP) is a very significant problem in computational biology as the secondary structure of protein is the key to other higher spatial structure of protein and as such the functionality of protein. Over the last six decades, researchers are trying to predict protein secondary structure by exploring newer method of prediction and using several features of protein. Success of using deep learning method in other fields in computer science motivated researchers of computational biology to incorporate this method in PSSP. In this paper, we summarized existing deep learning techniques in PSSP. We also proposed an ensemble method for PSSP which leverages different deep learning based predictors outcome. Our proposed approach of weighted voting seems to perform better than the individual deep learning based predictors. We execute extensive experiments to prove the accuracy of our method. In future, we want to incorporate larger datasets in our training process to improve the prediction accuracy further.

References

[1] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

[2] Harold A Scheraga. Structural studies of ribonuclease. iii. a model for the secondary and tertiary structure1, 2. *Journal of the American Chemical Society*, 82(15):3847–3852, 1960.

[3] AV Finkelstein. Ptitsyn, 0. b.(1971), statistical analysis of the correlation among amino acid residues in helical, beta-structural and non-regular regions of globular proteins. *J. Mol. Biol*, 62:613–624.

[4] Peter Y Chou and Gerald D Fasman. Prediction of protein conformation. *Biochemistry*, 13(2):222–245, 1974.

[5] Jean Garnier, David J Osguthorpe, and Barry Robson. Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins. *Journal of molecular biology*, 120(1):97–120, 1978.

[6] VI Lim. Algorithms for prediction of α-helical and β-structural regions in globular proteins. *Journal of Molecular Biology*, 88(4):873–894, 1974.

[7] Sujun Hua and Zhirong Sun. A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach1. *Journal of molecular biology*, 308(2):397–407, 2001.

[8] Jonathan J Ward, Liam J McGuffin, Bernard F. Buxton, and David T. Jones. Secondary structure prediction with support vector machines. *Bioinformatics*, 19(13):1650–1655, 2003.

[9] Zafer Aydin, Yucel Altunbasak, and Mark Borodovsky. Protein secondary structure prediction for a single-sequence using hidden semi-markov models. *BMC bioinformatics*, 7 (1):178, 2006.

[10] Xin-Qiu Yao, Huaiqiu Zhu, and Zhen-Su She. A dynamic bayesian network approach to protein secondary structure prediction. *BMC bioinformatics*, 9(1):49, 2008.

[11] Yan Liu, Jaime Carbonell, Judith Klein-Seetharaman, and Vanathi Gopalakrishnan. Comparison of probabilistic combination methods for protein secondary structure prediction. *Bioinformatics*, 20(17):3099–3107, 2004.

[12] Tom M. Mitchell. Artificial neural networks. In Tom M. Mitchell, editor, *Machine Learning*, chapter 4. McGraw-Hill Education.

[13] Burkhard Rost. Protein secondary structure prediction continues to rise. *Journal of structural biology*, 134(2-3): 204–218, 2001.

[14] Sheng Wang, Jian Peng, Jianzhu Ma, and Jinbo Xu. Protein secondary structure prediction using deep convolutional neural fields. *Scientific reports*, 6:18962, 2016.

[15] Sheng Wang, Wei Li, Shiwang Liu, and Jinbo Xu. Raptorx-property: a web server for protein structure property prediction. *Nucleic acids research*, 44(W1):W430–W435, 2016.

[16] Yangxu Wang, Hua Mao, and Zhang Yi. Protein secondary structure prediction by using deep learning method. *Knowledge-Based Systems*, 118:115–123, 2017.

[17] Matt Spencer, Jesse Eickholt, and Jianlin Cheng. A deep learning network approach to ab initio protein secondary structure prediction. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 12(1):103–112, 2015.

[18] Rhys Heffernan, Kuldip Paliwal, James Lyons, Abdollah Dehzangi, Alok Sharma, Jihua Wang, Abdul Sattar, Yuedong Yang, and Yaoqi Zhou. Improving prediction of secondary structure, local backbone angles, and solvent accessible surface area of proteins by iterative deep learning. *Scientific reports*, 5:11476, 2015.

[19] Qian Jiang, Xin Jin, Shin-Jye Lee, and Shaowen Yao. Protein secondary structure prediction: A survey of the state of the art. *Journal of Molecular Graphics and Modelling*, 76:379–402, 2017.

[20] Jinmiao Chen and Narendra S Chaudhari. Bidirectional segmented-memory recurrent neural network for protein secondary structure prediction. *Soft Computing*, 10(4): 315–324, 2006.

[21] Jinmiao Chen and Narendra Chaudhari. Cascaded bidirectional recurrent neural networks for protein secondary structure prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 4(4):572–582, 2007.

[22] Sepideh Babaei, Amir Geranmayeh, and Seyyed Ali Seyyedsalehi. Protein secondary structure prediction using modular reciprocal bidirectional recurrent neural networks. *Computer methods and programs in biomedicine*, 100(3):237–247, 2010.

[23] Claudio Mirabello and Gianluca Pollastri. Porter, paleale 4.0: high-accuracy prediction of protein secondary structure and relative solvent accessibility. *Bioinformatics*, 29 (16):2056–2058, 2013.

[24] Rhys Heffernan, Yuedong Yang, Kuldip Paliwal, and Yaoqi Zhou. Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility. *Bioinformatics*, 33(18):2842–2849, 2017.

[25] Pierre Baldi, Søren Brunak, Paolo Frasconi, Giovanni Soda, and Gianluca Pollastri. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, 15(11):937–946, 1999.

[26] Guang-Zheng Zhang, De-Shuang Huang, YP Zhu, and Yi-Xue Li. Improving protein secondary structure prediction by using the residue conformational classes. *Pattern Recognition Letters*, 26(15):2346–2352, 2005.

[27] N Jing, B Xia, CG Zhou, and Y Wang. Protein secondary structure prediction methods based onrbf neural networks. In *Computational Methods*, pages 1037–1043. Springer, 2006.

[28] Zhen Zhang and Nan Jing. Radial basis function method for prediction of protein secondary structure. In *Machine Learning and Cybernetics, 2008 International Conference on*, volume 3, pages 1379–1383. IEEE, 2008.

[29] Alexey Drozdetskiy, Christian Cole, James Procter, and Geoffrey J. Barton. Jpred4: a protein secondary structure prediction server. *Nucleic Acids Research*, 43(W1):W389–W394, 2015.

[30] James A Cuff and Geoffrey J Barton. Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins: Structure, Function, and Bioinformatics*, 40(3):502–511, 2000.