Secondary Protein Structure Prediction

Final Report for Team 3: *The Dexterz*

Sidharth Malhorta
Department of Computer Science
Northeastern University
360 Huntington Ave., Boston, MA 02115
malhotra.si@husky.neu.edu
UID: 001858737

Robin Walters
Department of Mathematics
Northeastern University
360 Huntington Ave., Boston, MA 02115
r.walters@northeastern.edu
UID: 001671191

Abstract

In this project we experiment with using neural network structures to predict a protein's secondary structure (α-helix positions) from only its primary structure (amino acid sequence). We implement a fully connected neural network (FCNN) and preform three experiments using that FCNN. Firstly, we do a cross-species comparison of models trained and tested on mouse and human datasets. Secondly, we test the impact of varying the length of protein sequence we input into the model. Thirdly, we compare custom error functions designed to focus on the center of the input window. At the end of paper we propose a alternative, recurrent neural network model which can be applied to the problem.

1 Introduction

1.1 Background: Protein Structure

Proteins are large molecules in living cells which perform many different roles, for example, catalyzing biochemical reactions, signaling across the cell, and transmitting and verifying genetic information. This makes them one of the most essential molecules for living organisms [2].

Chemically, proteins are formed by long chains of small molecules called amino acids bonded together. There are 20 different kinds of amino acids commonly found in eukaryotes. The amino acid sequence is the protein's *primary structure*. The chemical and physical properties of the different amino acids, together with the flexibility of the bonds between them, cause the long chain to fold into various shapes, the most common of which are α-helices, β-sheets and β-turns. These shapes constitute the protein *secondary structure*. In turn, these secondary structures fold and refold into more complex 3D shapes, the *tertiary structure* of the protein. Lastly, when two or more of these polypeptide chains fold together, they form a *quaternary* structure [8, 7].

1.2 Problem

The structure of a protein determines its functionality within the cell and how it interacts with other molecules such as drugs. It is thus very important to determine the 3D structure of a protein with high accuracy. Learning the primary structure, or sequence, of a protein is relatively cheap and easy, but finding the full folded structure is expensive and difficult, requiring lab techniques such a nuclear magnetic resonance imaging and X-ray diffraction crystallography. Consequently, the percent of structures known among all known protein sequences is only about 0.2%. The recognized functions are even less. This poses a big challenge for biologists trying to understand the structures and functions of proteins [2].

In this paper, we address the problem of predicting folded structure using only the sequence, with the aim of providing a faster and cheaper alternative to lab-based methods. Specifically, the problem we are trying to solve in this paper is as follows:

Problem: *Given as input only a protein's primary structure, its amino acid sequence, predict part of its secondary structure, which acids in the sequence are parts of an α-helix.*

CS6410 Final Project, Professor Rose Yu (Fall 2018), Northeastern University, Boston, Massachusetts.

The ultimate goal is to be able to predict the complete secondary, tertiary, and quaternary structure from the sequence alone, but this is currently out of reach. Nonetheless, the study of secondary protein structures is considered a necessary and important milestone to predict the 3D structure of a protein and can further allow us to understand the complex functionalities of a protein [11].

1.3 Challenges

This problem is hard for several reasons. Protein sequence lengths can be over 1000 amino acids long and with 20 different possible acids per position, the number of sequences is enormous. This necessitates some method to limit input size such as using only small sections of the sequence, *windows*, at a time. Moreover, protein structure depends subtly on sequence. Changing or mutating even a single amino acid can change the overall structure dramatically. Lastly, *a priori*, acids distant from each other in the sequence can still interact in the folded structure, making it difficult to limit the length of window too greatly.

1.4 Contributions

In this paper, we implement and train different machine learning models to predict the secondary structures of proteins. We compare and discuss the results from deep learning models like fully connected neural networks (FCNN) and sequential models like recurrent neural networks (RNN). We also perform three experiments using our models to help with protein structure classification. Here is a summary of our contributions to the problem:

- Cleaned and removed redundancy from a mouse and human protein dataset. Built code to import PDB files into PyTorch.
- Built and trained two models, a fully-connected neural network (FCNN), and a recurrent neural network (RNN). The FCNN has a good prediction root mean squared error (RMSE) of 0.22.
- Performed a cross-species comparison by training two FCNNs, one on a mouse protein dataset and one on a human protein dataset. We then tested both models on both the mouse and human test sets. The models performed best on their own species test sets, but did very well on the cross-species test as well. This validates the concept of using mice as model organisms within this context.
- Tested different window sizes of 7, 10, and 13 amino acids. Despite our hypothesis that 10 or 13 would perform best, 7 had the lowest loss. The implication is that at least for α-helix prediction, sequence length and input size are not large challenges.
- Tested different error functions such as Gaussian, unweighted RMSE, and centered. We found unweighted and Gaussian performed best.

2 Related Work

Yang [10] compared algorithmic efficiencies and computational times for single sequence protein prediction through neural network and support vector machines based algorithms. Yang tried different window and hidden layer sizes from the range 1 to 21 and 0 to 125 respectively. The neural network approach peaked with a performance of 67.42% accuracy at a window size of 15 amino acids and a hidden layer of 75 units. Although, SVMs performed better at convergence than neural networks and did not tend to overfit, the overall performance of neural network outperformed that of the SVMs.

Malekpour [5] improved the existing method of Segmental semi Markov models (SSMMs) using three neural networks that used multiple sequence alignment profiles. The outputs of the neural networks were passed through SSMMs to predict secondary structures of amino acids. The proposed model predicted the protein structures with an overall accuracy of 75.35%.

Jones [3] attempted a PSIPRED method that achieved an accuracy score of between 76.5% to 78.3%. PSIPRED works on position specific scoring matrices. The method is split into three main stages: generation of sequence profile, prediction of initial secondary structure, and filtering of the predicted structure.

King [4] mentions about PROMIS, a machine learning program that predicted secondary structures in protein up to the accuracy of 60%, using the generalized rules that characterize the relationship between primary and secondary structure in globular proteins.

Pollastri [6] uses an ensemble of bidirectional RNNs to predict the protein secondary structures in three and eight categories resulting in deriving two new predictors. The predictors tested on three different test sets perform with an accuracy of 78%.

3 Method/Model

3.1 Feature Extraction and Engineering

The protein data bank (pdb) files that we selected for our dataset contained much information about each protein including the complete 3D structure of the protein, the amino acid sequence, and the location of various secondary structure such as α-helices. We extracted helix structure information from all the pdb files and then used one-hot encoding to represent each amino acid type, and binary values (1 or 0) to represent our target variables (helix structures) in a protein sequence. There can be 20 different types of amino acids in a protein sequence which resulted in one-hot encoded vectors of length 20 each. [9]

3.2 Fully Connected Model

We selected a fully connected neural network (FCNN) with 200 input neurons, 40 hidden neurons, and 10 output neurons as our first training model. This type of model is also called a multilayer perceptron (MLP). The input vector represents a window of 10 sequential amino acids in the protein sequence. We encode each type of amino acid using a standard basis vector $\mathbf{e}_i$ in $[0,1]^{20}$ and then flatten this into an input vector $\mathbf{x}$ of length 200. Define the activation function RELU by

$$\mathrm{R}(x) = \begin{cases} x, & \text{if } x \geq 0, \\ 0, & \text{if } x < 0. \end{cases}$$

Then define the hidden layer values as

$$\mathbf{h}_i = \mathrm{R}\left(\sum_{j=1}^{200} w_{i,j}^{(1)} \mathbf{x}_j\right)$$

where $w_{i,j}^{(k)}$ are the models trainable weights.

The output vector is then defined

$$\mathbf{o}_i = \mathrm{R}\left(\sum_{j=1}^{40} w_{i,j}^{(2)} \mathbf{h}_j\right).$$

We interpret these ten values as the predicted probability of finding a helix (or *helicity*) at each of the 10 positions within the window. We can reconstruct a prediction for the entire protein by moving this window over the entire sequence and extracting the middle values (or as close as possible) at each position.

Such a model can be thought of as analogous to a one-dimensional convolutional model in which the input and output vectors stretch over the entire protein sequence since we use the same shared weights across the entire sequence. Figure 1 is a diagrammatic version of our model.

4 Experiment

4.1 Training Error Functions

For the purpose of this paper, we implemented a total of three error functions, namely, unweighted loss, Gaussian loss and centered loss (See Figure 2). All loss functions are weighted RMSE functions defined as such

$$E(\overline{p}, \overline{y}) = \sqrt{\sum_{i=1}^{m} w_i (p_i - y_i)^2},$$

where i indexes over the window of length m, $\overline{y}$ are the true values, $\overline{p}$ are the predicted values and $\overline{w}$ is the weight vector.

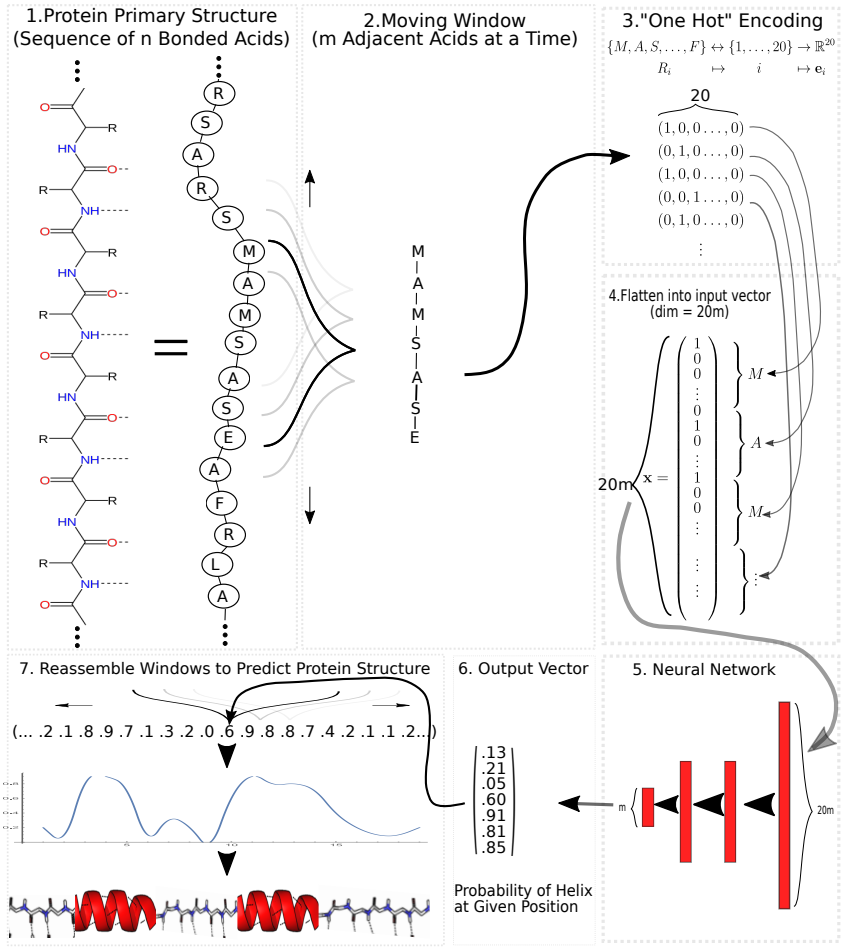

Figure 1: Diagram of the FCNN.

The unweighted error function uses $w_i = 1/m$ and is equivalent to root mean squared error (RMSE). The Gaussian weight is defined

$$u_i = e^{\left(\frac{(i-m)^2}{5m}\right)^{4/3}} \tag{1}$$

$$w_i = u_i / W \tag{2}$$

where $W = \sum_{i=1}^{m} u_i$. The centered loss is defined

$$w_i = \begin{cases} 1 - \frac{m-1}{100}, & \text{if } i = \lfloor \frac{m}{2} \rfloor \\ \frac{1}{100}, & \text{otherwise.} \end{cases} \tag{3}$$

In the unweighted loss function, all the acids in a single window of protein sequence have equal weight, whereas in Gaussian loss, the weight is greatest for the acid at the center of the window and decreases gradually towards the ends. At the extreme, for centered loss, only the acid at the center has significant weight whereas all other acids in the window have negligible weight. We selected these functions to test our hypothesis that in a moving window parsing of a protein sequence, acids at the center of the window should play a major role in predicting the accurate location of the helix structure in the protein sequence and should have better predictions.

4.2 Test Criterion

The above error functions are for training the models. They are give the error between the true helicity and predicted helicity on a short window of the sequence. For testing, it is necessary to use a standard

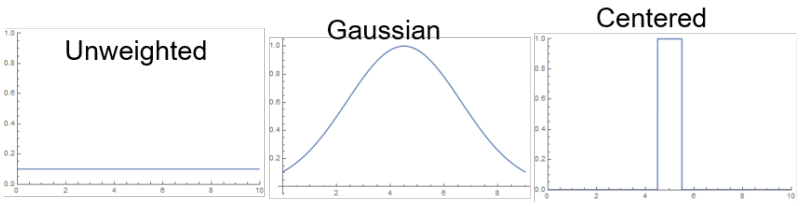

Figure 2: Different Loss functions

error function to compare the results. We use standard RMSE *over the length of the entire protein* as our test criterion. In this case, the predicted values of helicity are reconstructed by moving the window over the entire protein sequence and extracting a helicity predictions for each position using the window in which it is closest to the center. *Any references to average test loss throughout refer to this definition.*

4.3 Dataset

The Protein Data Bank (PDB) is a very large and diverse dataset with proteins from different species and varying lengths and structures. Figure 3 and Figure 4 represent the variability of the dataset [1].

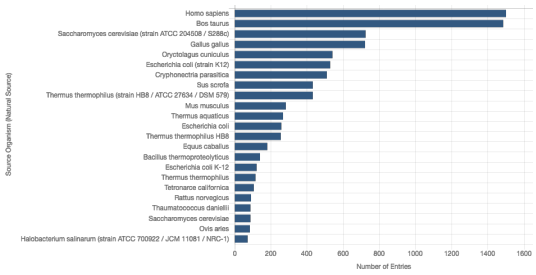

Figure 3: Number of proteins in PDB by species.

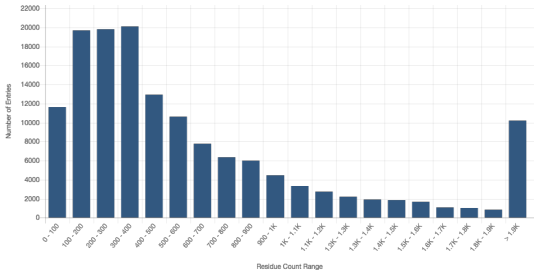

Figure 4: Number of proteins in PDB by length.

It is likely that proteins of drastically different sizes and proteins separated by billions of years of evolution rely on different sequence motifs for their structures. In fact, early tests in which we pulled proteins from the databank randomly fit very poorly. Thus, to address this challenge we created species-specific datasets for just two closely related animals, mouse and human, in order to increase similarity and patterning in the set. Due to the large size of the initial dataset, these subsets were still very large.

We collected structural data of 30,000 human proteins and 6,000 mouse proteins. We then subdivided each species-specific dataset, randomly assigning 80% of our data as training data and 20% of our data as test data. Many proteins are very symmetric including several sequence-identical chains. In these cases, we used only one copy of each unique chain in order to reduce redundancy in the dataset.

4.4 Baselines

We used a simple baseline against which we compared our more sophisticated models. The helix-probability of a given amino acid type R is defined

$$P(\text{Helix}|R) = H_R/N_R,$$

where N_R is the number of positions in the dataset which are of type R. Let H_R be the number of positions which are in a helix and of type R. The helix probabilities for the 20 amino acids are shown in Figure 5.

This gives baseline probability p_i that the position i is part of helix given that it is of amino acid type R. We can thus use the vector $(p_i)_i$ as a baseline prediction of helicity. Using this baseline on the mouse dataset gives an average loss of 0.48. Any machine learning method which is not beating this rudimentary statistical approach is failing.

Amino Acid	Helix Prop	Amino Acid	Helix Prop
ALA	43.1%	GLU	41.7%
GLY	27.9%	ARG	37.6%
ILE	38.1%	HIS	33.9%
LEU	42.2%	LYS	37.4%
PRO	27.0%	SER	31.7%
VAL	33.5%	THR	31.9%
PHE	38.4%	CYS	32.3%
TRP	38.8%	MET	43.2%
TYR	35.3%	ASN	31.6%
ASP	35.2%	GLN	40.3%

Figure 5: Table of helix-probabilities for each acid.

A slightly more sophisticated baseline can be made by using sequence of 2 or 3 amino acid types such as

$$P(\text{Helix}|R_1R_2R_3).$$

Another basic score which can be used similarly is the helix-propensity of a given amino acid type R defined

$$P_R = \frac{P(\text{a position in a helix is type R})}{P(\text{a position is type R})}.$$

4.5 Model Implementation and Training

The implementation of our model can be found here: `https://github.com/sidharth0094/protein-structure-prediction`

Our FCNN trained on the mouse dataset had an average test loss of 0.21. We include several graphs showing the outputs in Appendix A.

We tried two different approaches to implement and train the FCNN presented in Section 3. The first version of the model we implemented in Keras and trained by randomly shuffling protein sequence windows (each of size 10) in every epoch. Whereas the second model we implemented in Pytorch and trained differently. In each epoch we shuffled the order of the proteins, but kept the sequence windows in order within each protein.

The first model in Section 4.5 had a very high average loss of 0.6. In Figure 6, we compare predicted helix probability to actual helix presence. The prediction (blue line) is close to random noise around the mean. However, the second model in Section 4.5, predicted with average loss of 0.21 (Figure 7).

We conjecture the second model preformed better in part due the way in which we trained it. In both neural network models, we randomly sampled proteins, and shuffled the order in each epoch. However, in the second model we ran through all the windows of each protein we selected *in order*. Thus, due to the overlapping windows, the model trained on each animo acid position several times, seeing it with slightly different context (surrounding sequence) each time.

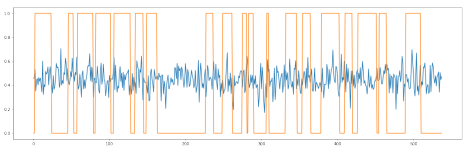

Figure 6: First Training Run.

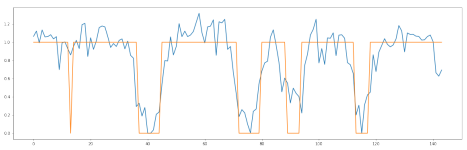

Figure 7: Second Training Run.

Figure 8: The x-axis is the position index in the protein sequence. The y-axis is probability of a helix (helicity). The orange line shows the true presence of helices. The blue line is the model's prediction of the probability of a helix.

4.6 Cross Species Comparison

We trained our FCNN model on random subsets of the mouse training set and human dataset of equal size (5000 proteins). We then took each model and tested them on both their own species' test set and on the other species test set. The results are illustrated in Figure 9. As expected, we found that each model fit its own species best. The average RMSE for the human-trained human-tested model was 0.2177 and for mouse-trained mouse-tested it was 0.2279. However, each model also did surprising well in the cross-species test. The human-trained model had an average RMSE of 0.2485 on the mouse test set and mouse-trained model had an average RMSE of 0.2419 on the human test set.

We interpret these results as a validation, in this context, of the use of mice as a model species for humans in laboratory environment. The good cross-species fit implies that mouse and human proteins are largely very similar. Looking at the losses for specific proteins shows that the mouse and human trained models often performed similarly on many targets but that the cross-species loss was worsened by certain more species-specific proteins which did not have strong analogues in the other training set.

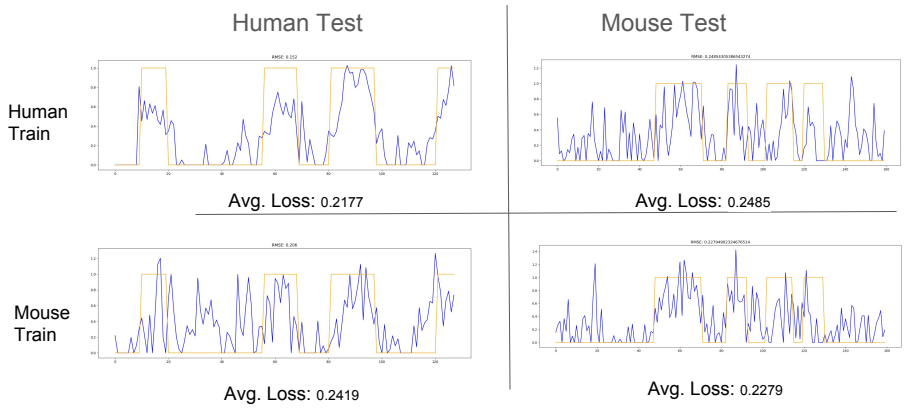

Figure 9: Cross Species Comparison. The Avg. Loss indicates the average RMSE over the entire test set, not just the example.

4.7 Error Function Comparison

We trained our FCNN model on the mouse training set using the different error functions mentioned in subsection 4.1 and tested them on the mouse test set. The results are illustrated in Figure 10. Surprisingly, we found that the unweighted RMSE and Gaussian error performed almost the same with the average loss of 0.2280 and 0.2287 respectively. However, the centered error resulted in an average loss of 0.2330. The results from testing different error function contradicted our hypothesis that giving more weight to the amino acid at the center of the window would result in better prediction of the location of secondary structures.

4.8 Window Size Comparison

We conducted an experiment using our FCNN model on the mouse dataset with one hidden layer of 40 neurons and window sizes of 7, 10, 13. We selected these windows because each turn of the α-helix takes 3.5 amino acids and thus these sizes correspond to 2, 3, and 4 turns respectively. Our Hypothesis was that a window size of 10 or 13 should outperform the shorter window in predicting the accurate secondary structures since more information is available at longer lengths.

The results are illustrated in Figure 11. The window size of 7 resulted in the average loss of 0.1933, whereas the window sizes of 10 and 13 resulted in average losses of 0.2280 and 0.2590 respectively. These results contradicted our hypothesis that a window size of 10 or 13 would have a lower loss since larger windows sizes give the model greater context to predict structure. Instead, it was the smallest window size which resulted in the best prediction. It is likely 7 is close to the smallest size which would give good results. A window size of 1, for example, implies no context and is essentially equivalent to using the baseline, which had average loss of 0.48.

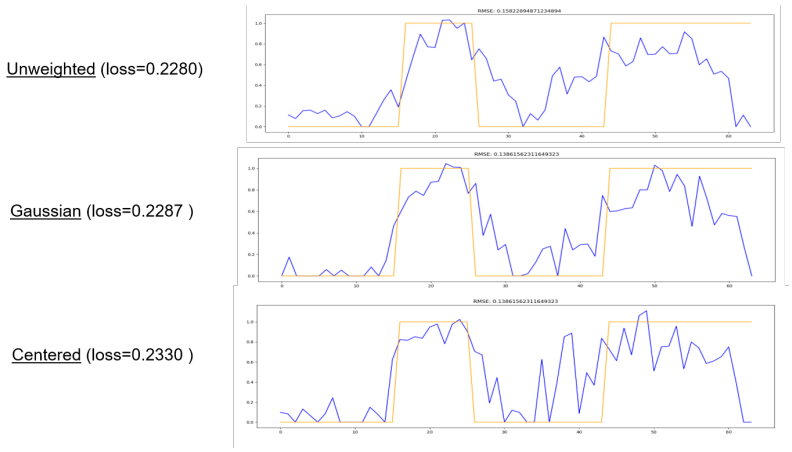

Figure 10: Error Function Comparison. The Avg. Loss indicates the average RMSE over the entire test set, not just the example.

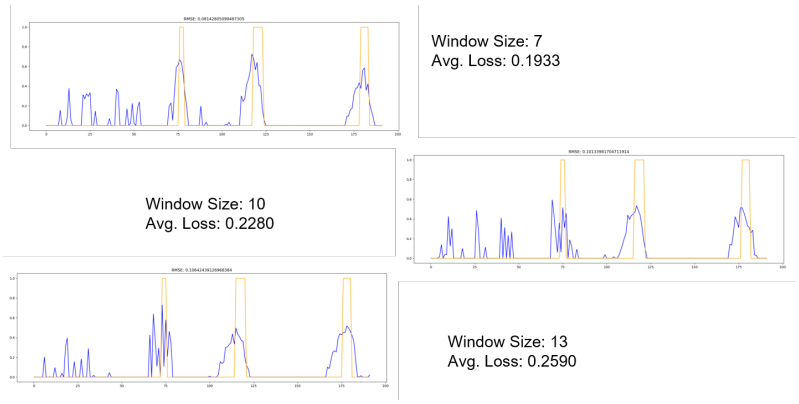

Figure 11: Window Size Comparison. The Avg. Loss indicates the average RMSE over the entire test set, not just the example.

5 Next Steps: Recurrent Neural Network

We implemented a recurrent neural network which takes in a single amino acid at a time. Let $\mathbf{x}^{(i)}$ be the length 20 encoding of the i^{th} amino acid. Then define $h^{(0)} = \mathbf{0}$ and $h_i^{(k)} = R\left(\sum_{j=1}^{20} w_{i,j}^{[1]} x_j^{(k)} + \sum_{j=1}^{40} w_{i,20+j}^{[1]} h_j^{(k-1)}\right)$ and $\mathbf{o}_i^{(k)} = R\left(\sum_{j=1}^{40} w_{i,j}^{[2]} h_j^{(k)}\right)$. We can visualize this network as in Figure 12.

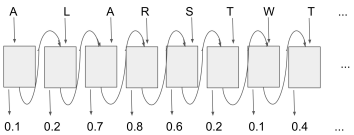

Figure 12: Diagram of the RNN.

We wish to train this recurrent neural network and compare the error to the FCNN. Since, protein chains are sequential in nature, we believe that an RNN should be able to capture the patterns in the dataset better than a FCNN.

References

[1] BERMAN, H. M., WESTBROOK, J., FENG, Z., GILLILAND, G., BHAT, T. N., WEISSIG, H., SHINDYALOV, I. N., AND BOURNE, P. E. The protein data bank. *Nucleic Acids Res 28* (2000), 235–242.

[2] JIANG, Q., JIN, X., LEE, S.-J., AND YAO, S. Protein secondary structure prediction: A survey of the state of the art. *Journal of Molecular Graphics and Modelling 76* (2017), 379–402.

[3] JONES, D. T. Protein secondary structure prediction based on position-specific scoring matrices 1 1edited by g. von heijne. *Journal of Molecular Biology 292*, 2 (1999), 195–202.

[4] KING, R. D., AND STERNBERG, M. J. Machine learning approach for the prediction of protein secondary structure. *Journal of molecular biology 216*, 2 (1990), 441–457.

[5] MALEKPOUR, S. A., NAGHIZADEH, S., PEZESHK, H., SADEGHI, M., AND ESLAHCHI, C. Protein secondary structure prediction using three neural networks and a segmental semi markov model. *Mathematical Biosciences 217*, 2 (2009), 145–150.

[6] POLLASTRI, G., PRZYBYLSKI, D., ROST, B., AND BALDI, P. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *Proteins: Structure, Function, and Bioinformatics 47*, 2 (2002), 228–235.

[7] SERVICES, P. S. D. D. Protein structure. *Technical Breif 8* (2009).

[8] WIKIBOOKS. Structural biochemistry/proteins — wikibooks, the free textbook project, 2018. [Online; accessed 22-November-2018].

[9] WIKIPEDIA CONTRIBUTORS. Proteinogenic amino acid — Wikipedia, the free encyclopedia, 2018. [Online; accessed 24-November-2018].

[10] YANG, J. Protein secondary structure prediction based on neural network models and support vector machines. *Departments of Electrical Engineering, Stanford University* (2008).

[11] YOO, P. D., ZHOU, B. B., AND ZOMAYA, A. Y. Machine learning techniques for protein secondary structure prediction: an overview and evaluation. *Current Bioinformatics 3*, 2 (2008), 74–86.

A Example Predictions from the Mouse FCNN

We present some illustrations of the predictions made by the FCNN on the mouse test set. In each graph, the x-axis is the position index in the protein sequence. The y-axis is probability of a helix. The orange line shows the true presence of helices. The blue line is the models prediction of the probability of a helix.

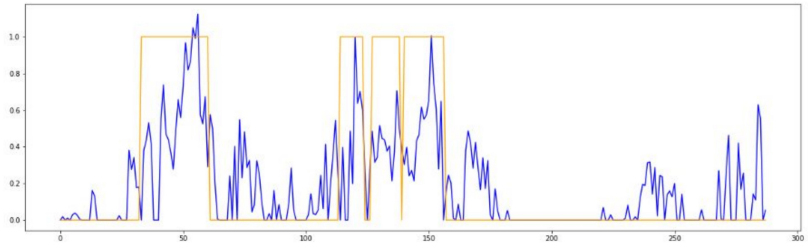

Figure 13: Protein 1CD1. Loss = 0.15. A very good result. Note the very strong dip in the blue line between the second and third helix.

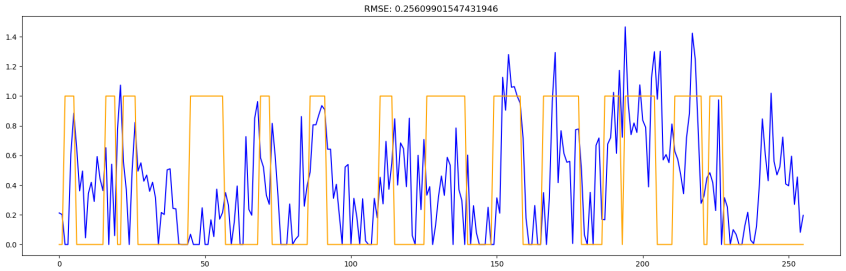

Figure 14: Protein 1AQU. Loss = 0.256. A medium-quality result. Many helices are clearly marked by spikes, but the beginning and ending points are unclear. Some helices are missed outright and small valleys between helices are indistinct.

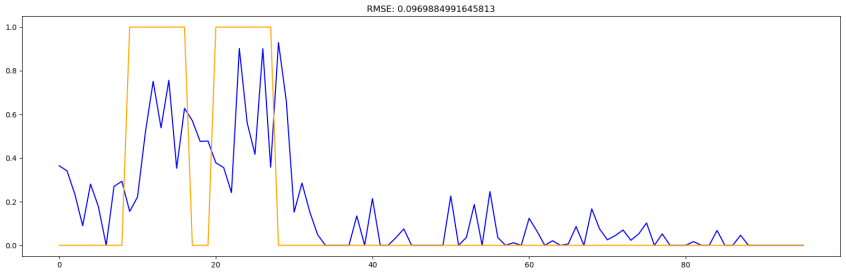

Figure 15: Protein 1AB0. Loss = 0.097. A good result. Clearly detects both helices and their lengths but errs in predicting their positions slightly too late in the sequence. This result also does a good job of predicting no helices through the rest of the sequence.

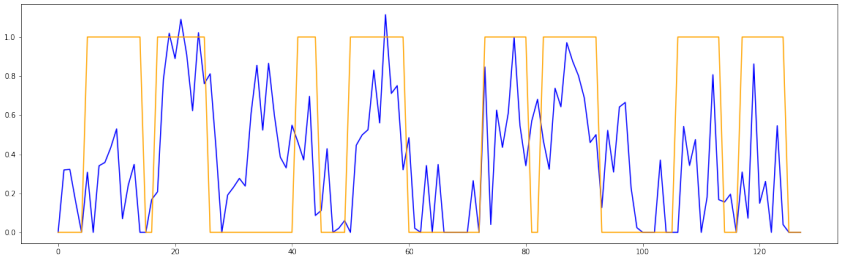

Figure 16: Protein 1AP7. Loss = 0.2462

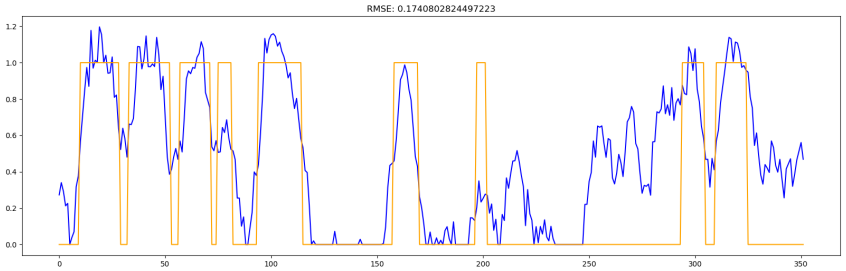

Figure 17: Protein 1DD7. Loss = 0.1741