Protein Secondary Structure Prediction with Deep Learning Neural Networks

Burhan Ariyanto, Lujie Yu, Sanura Devmin Amarasekara
M2 MIAGE, Université Toulouse 1 Capitole,
Toulouse, France
Burhan.Ariyanto@ut-capitole.fr
Lujie.Yu@ut-capitole.fr
Sanura-Devmin.Sanura@ut-capitole.fr

Abstract

Protein structure prediction is one of the crucial issues in present day computational biology. The primary structure of a protein is its amino acid sequence and the secondary structure of a protein can be used to predict the tertiary structure. Alpha (α) helix and beta (β) sheet are the most common secondary protein structures. The secondary structure prediction is a set of techniques in bioinformatics that aims to predict the local secondary structure of protein. There are many emerging methods (supervised or unsupervised machine learning) to address the problem of protein secondary structure prediction (PSSP). In this paper, we will review the use of deep learning neural networks, as they are powerful methods for studying such large data sets and has shown superior performance in many areas of machine learning. We used the development and application on deep learning neural networks, which are CNN and GCN, to predict the secondary structure of protein using the amino acid sequences as inputs. Our results confirm that the presence of amino acids in the protein sequence increases the stability for the approximation of the secondary structure of the protein.

Keywords: *Secondary structure (SS), Prediction, Q3-state, Q8-state, Protein, Amino acid, deep learning, Neural networks, CNN, GCN.*

1 Introduction

Proteins are large and complex substances that play many important roles in all living organisms. They are the main actors in cells, from DNA repair to enzyme catalysis and they are required for the structure, function, and regulation of body's tissues and organs, which include important biological compounds such as enzymes, hormones and antibodies. They are made up of thousands of smaller units called amino acids, which are bonded to each other in long chains.

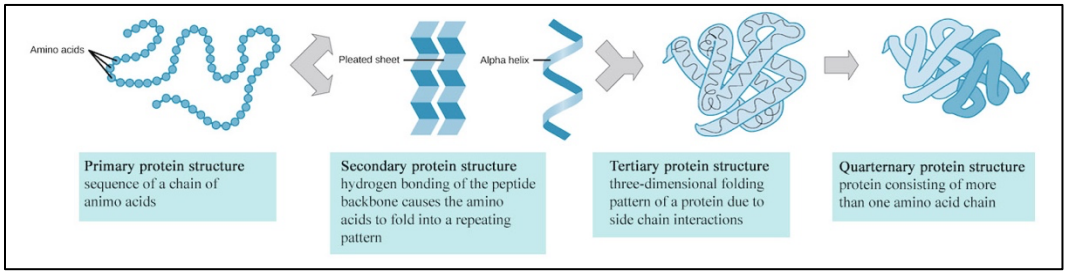


Figure 1: The four different levels of protein structure.

The biological function of protein is determined by the arrangement of atoms in a three-dimensional structure. The notorious "spike protein" that binds to the coronavirus allows the virus to enter our cells. Meanwhile, mRNA vaccines such as Pfizer and Moderna, mimic the shape of the spike protein that causes the body to produce an immune response. However, determining protein structure (via experimental techniques such as X-ray crystallography, nuclear magnetic resonance, and cryo-electron microscopy) was difficult, slow, and expensive. Having protein structure is important because it can provide a greater level of understanding of how proteins work, which allows us to make hypotheses about how to influence, control, or modify them. It can be applied in biological processes such as drug and/or enzyme design [1], antibody development and interpretation of mutations in structural genomics. There are four levels of protein structure, which are primary, secondary, tertiary, and quaternary (see figure 1), which can help us to understand the nature and function of each level of protein.

One of the greatest challenges in computational biology is understanding the complex sequence-structure relationship, therefore accurate prediction of protein structure relies on the accuracy of secondary structure prediction. The development of machine learning (ML) and deep learning (DL) methods has made this prediction more accurate. The application of deep learning in bioinformatics to gain insights from data has been emphasized in various fields. Deep neural networks (DNN) are very popular for predicting protein structure because they excel at solving problems where there are complex relationships between input features and desired outputs.

To tackle the difficulties in predicting secondary structure, the use of deep neural networks (DNN), with a lot of layers, and train deep neural architectures based on the amino acids sequence is applied. Deep neural networks are subset of artificial neural networks (ANN) with many layers between the input and output layers. There are different types of neural networks (i.e., ANN, CNN, RNN, etc.) but they have things in common that they always consist of the same components: neurons, weights, biases, and functions.

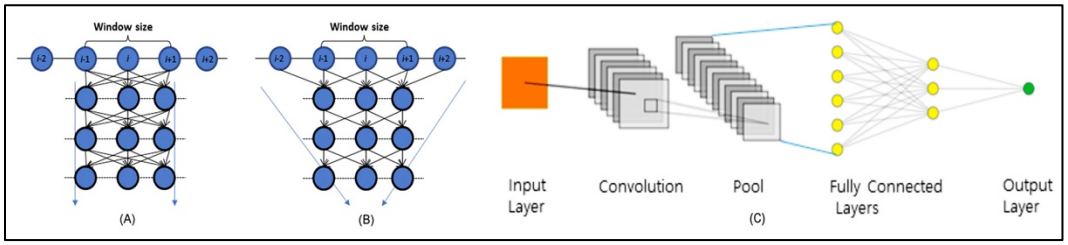


Figure 2: A typical model of deep neural network (A) vs. convolutional deep neural network (B). and an overview over CNN (C) that can capture longer-range sequence information and have the ability to express various functions and their efficiency depending on the amount of quality data.

2 State of the Art

2.1. Protein Structure

Amino acids are the building blocks of proteins in all organisms. There are more than 500 amino acids found in nature, but the human genetic code only codes for 20. From these twenty amino acids can be divided into two groups: essential and non-essential. Non-essential amino acids are those that the human body is able to synthesize, whereas essential amino acids must be obtained from food during illness or as a result of health problems. Non-essential amino acids are alanine, arginine, asparagine, aspartate, cysteine, glutamic acid, glutamine, glycine, proline, serine and tyrosine. The essential amino acids are histidine, isoleucine, leucine, lysine, methionine, phenylalanine, threonine, tryptophan, and valine.

Regarding the 20 natural amino acids in the human body, they are denoted by one letter notation: 'A', 'C', 'D', 'E', 'F', 'G', 'H' , 'I', 'K' ', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y'. 'A' stands for Alanine, 'C' for Cysteine, 'D' for Aspartic Acid, 'E' for Glutamic Acid, etc. Whereas the 21st letter, 'X', is sometimes used to denote an unknown amino acid.

As explained in the introduction, a protein secondary structure is important, because it can be used to predict its tertiary structure. Protein structure prediction is effectively used to define 3D protein structures that support more genetic information since many experimental biologists experience the limited availability of 3D protein structures.

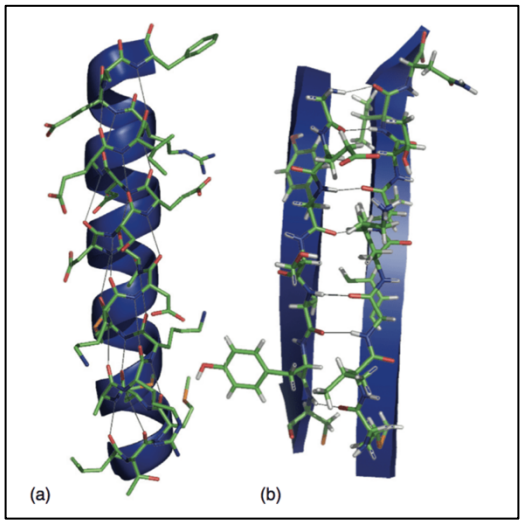


Figure 3: α helix (a) & β sheet (b) of protein secondary structure.

When predicting the secondary structure of a protein, we distinguished between a 3-state prediction (Q3) and an 8-state prediction (Q8). The goal for 3-state prediction is to classify each amino acid into α-helix (H), β-strand (E) and coil region (C) (see figure 3). Meanwhile for 8-state prediction classifies each amino acid into 3 types for helixes (G , H, and I), 2 types for strand (E and B) and 3 types for coil (T, S, and L) (see table 1).

Table 1: The 8-state to 3-state mapping of protein secondary structure

8-class	3-class	Name
H	H	α-helix
E	E	β-strand
L	C	loop or irregular
T	C	β-turn
S	C	bend
G	H	3_{10}-helix
B	E	β-bridge
I	C	π -helix

2.2. Related Works

The inauguration of secondary structure of proteins was initiated by Pauling and Corey [2] in 1951, to provide an approximate picture of the overall structural category. This poor prediction relied on training large datasets leading to overfitting and inability of classifiers to estimate unknown datasets. Sander and Kabsch [3] developed a DSSP (Dictionary of Secondary Structures in Proteins) algorithm to standardize secondary structure assignments. This was the first method for the determination of the secondary structure of proteins available as a computer program, and remains

the most popular today. DSSP classifies each amino acid residue in proteins with a known 3D structure into 8 fine-grained states, based on the recognition of hydrogen bonding patterns.

The neural networks (NNs) first were used by Qian & Sejnowski in 1988 [4], followed by Holley and Karplus [5] to predict secondary structure. Utilizing evolutionary information using sequence profiling of multiple alignments (Rost & Sander, 1993 [6]), has significantly improved results for secondary structure prediction, or by utilizing the position-specific scoring matrix of PSI-BLAST (Altschul et al., 1997 [7]). Other significant developments include the use of bidirectional recurrent neural networks (BRNN) to better capture spatial dependencies (Baldi et al., 1999 [8]), and the use of probabilistic graphical models (Schmidler et al., 2000 [9]; van der Maaten et al., 2011 [10]). Deep learning methods have also been applied due to the much more complicated prediction of Q8. For example, the use of SC-GSN network, the use of bidirectional long short-term memory (BLSTM) method [11], and the next stage conditioned convolutional neural network (CNN) [12].

In the bioinformatics literature, the two most widely used algorithms for protein secondary structure prediction are PSIPRED by Jones in 1999 [13] and Jpred by Drozdetskiy in 2015 [14]. Jones developed the a 2-stage neural network method, which uses the PSI-BLAST sequence profile as input and obtains ~80% accuracy for prediction of 3-state SS. Whereas the Jpred model which uses the multilayer perceptron (MLP) structure and considers more features-based approach with multiple models, claiming a Q3 score of 81.5%. Qi et al. in 2012 [15] used a deep MLP architecture with multitasking learning and achieved 81.7% on Q3 prediction. Zhou and Troyanskaya in 2014 [16] created a generative stochastic network to predict the secondary structure for Q8 by 66.4%.

Table 2: Method comparison on Q8 prediction problem by using newly released structures (TS115) and an encoded protein gene (CASP12)

Dataset Method	TS115		CASP12		Server location
	Q8	P-value*	Q8	P-value*	
SSPRO8	0.68	3E-9	0.69	0.014	http://scratch.proteomics.ics.uci.edu
DeepCNF	0.72	NA	0.73	NA	http://raptorx2.uchiago.edu/StructurePropertyPred/predict/

*Paired t-test from DeepCNF

For the scope of this project, our objective is to predict the sequence of SST8 (sst8 column of the dataset) based on the amino acid sequence (seq column) to tackle the problem of secondary structure prediction. We will review the use of deep neural networks (DNN) and train different deep neural architecture to achieve high accuracy on the prediction. For the first method, we propose to use convolutional neural networks (CNN) and we think the algorithm is suitable for predicting protein structure, which can label the properties of individual amino acids across the target sequence at once. Convolutional neural networks have local perceptual, down-sampling and weight-sharing characteristics, which propose that each neuron perceives only local pixels of the image, and then combines this local information at higher layers to obtain all image characterization information. The second method we use is graph convolutional networks (GCN). We have tried with recurrent neural network (RNN) and long-short term memory (LSTM) but the accuracy is lower than CNN.

3 Materials and Methods

3.1. Dataset

Based on the given dataset, it consists of 9078 proteins which have sequence lengths varying between 20 - 1632 and it is based on protein data bank (PDB). For testing and training purposes, we differentiate the dataset into original dataset and training dataset which we cleaned using numpy and panda from python. The training dataset will contain only 3 columns : seq, sst8 and sst3.

The given dataset (see figure 4) lists the peptide sequences and their corresponding secondary structures. There are 17608 non-redundant chains (25%). Here is the column description:

1. pdb_id: the id used to find the entry
2. seq: sequence of amino acids in a single letter code
3. sst3: three-state secondary structure (Q3)
4. sst8: assignment of 8-states (H,B,E,G,I,T,S,C) of secondary structure at each amino acid residue corresponding to the amino acid of the respective PDB sequence.

However, this dataset can be downloaded from this site [26] for a more detailed description.

	pdb_id	chain_code	seq	sst8	sst3	len	has_nonstd_aa	Exptl.	resolution	R-factor	FreeRvalue
0	1FV1	F	NPVVHFFKNIVTPRTPPPSQ	CCCCCBCCCCCCCCCCCCCC	CCCCCECCCCCCCCCCCCCC	20	False	XRAY	1.90	0.23	0.27
1	1LM8	H	DLDLEMLAPYIPMDDDFQLR	CCCCCCCCCBCCSCCCEECC	CCCCCCCCCECCCCCEECC	20	False	XRAY	1.85	0.20	0.24
2	1O06	A	EEDPDLKAAIQESLREAEEA	CCCHHHHHHHHHHHHHHHTC	CCCHHHHHHHHHHHHHHHCC	20	False	XRAY	1.45	0.19	0.22
3	1QOW	D	CTFTLPGGGGVCTLTSECI*	CCTTSCTTCSSTTSSTTCCC	CCCCCCCCCCCCCCCCCCCC	20	True	XRAY	1.06	0.14	1.00
4	1RDQ	I	TTYADFIASGRTGRRNAIHD	CHHHHHHTSSCSSCCCCEEC	CHHHHHHCCCCCCCCCEEC	20	False	XRAY	1.26	0.13	0.16
5	1T6O	B	QDSRRSADALLRLQAMAGIS	CHHHHHHHHHHHHHHHTCC	CHHHHHHHHHHHHHHHCCC	20	False	XRAY	2.00	0.23	0.28
6	1T7F	B	SSRGLLWDLLTKDSRSGSGK	CCCCHHHHHHCCCCCCCCCC	CCCCHHHHHHCCCCCCCCCC	20	False	XRAY	1.60	0.20	0.22
7	1U7B	B	SRQGSTQGRLDDFFKVTGSL	CCCCCCBCCGGGTSBCCCCC	CCCCCCECCHHHCCECCCCC	20	False	XRAY	1.88	0.22	0.27
8	1UGX	B	DEQSGISQTVIVGPWGAKVS	CCCCCSCCCEEEEEEECCCC	CCCCCCCCCEEEEEEECCCC	20	False	XRAY	1.60	0.19	0.20
9	1VPP	Y	RGWVEICAADDYGRCLTEAQ	CCCEEEEEBCTTSCBTTCCC	CCCEEEEEECCCCCECCCCC	20	False	XRAY	1.90	0.19	0.27

Figure 4: An overview of the given dataset of proteins.

3.2. Convolutional Neural Networks

Convolutional neural networks are current state-of-art architecture for image or text classification tasks. CNN is being used everywhere, be it for processing sequential data such as audio, time series, NLP and in this session we use this algorithm to predict the secondary structure. The term convolution on CNN refers to the mathematical combination of two functions to produce a third function and then it combines two sets of information. There are 3 types of convolution operations. 1D convolution (used where input is sequential such as text or audio); 2D convolution (used where the input is an image) and 3D convolution (used in 3D medical imaging or detecting events in video). CNN helps extract features from text/images that can assist in data processing for prediction, by extracting low-dimensional features, and then some high-dimensional features such as shapes.

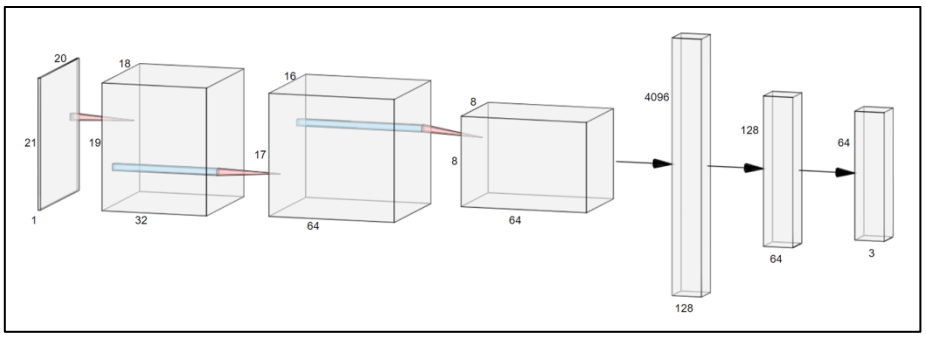


Figure 5: An overview of convolutional neural network of our model.

For the first method, we use a fairly simple convolutional neural network using a sliding window approximation on the sequence, with the PyTorch framework. The model that we develop consists of 3 main 2D convolution layers and it is usually abbreviated as conv2D (figure 5) with some drop-out and batch normalization followed by 3 fully connected (FC) layers. This nn.conv2D() applies 2d convolutional over the input and it will expect the input to be of the shape [batch_size, input_channels, input_height, input_width]. In our model, for example we wrote self.conv1 =

nn.Conv2d(1, 32, 3, 1). As a simplification, the output value of a layer with input size(N, C_{in}, H, W) and output (N, C_{out}, H_{out}, W_{out}) can be precisely described as:

$$\text{out}(N_i, C_{outj}) = \text{bias}(C_{outj}) + \sum_{k=0}^{C_{in}-1} \text{weight}(C_{outj}, k) \star \text{input}(N_i, k)$$

where N is the batch size, C represents the number of channels, H is the input field height in pixels, and W is the width in pixels.

The next modules we use are BatchNorm2D and ReLU. BatchNorm2D is the number of dimensions/channels that come out from the last layer and go into the batch norm layer. It is a technique that can increase the learning speed of neural networks by minimizing internal covariate shifts which are basically the phenomenon of changing the input distribution of each layer because the parameters of the layer above it change during training. BatchNorm2D is defined as follow:

$$y = \frac{x - \text{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}} * \gamma + \beta$$

where γ and β are learnable parameter vector of C (the input size). By default, the γ element is set to 1 and the β element is set to 0. The standard-deviation is calculated via the bias estimator, equivalent to *torch.var(input, unbiased=False)*.

nn.ReLU is an activation function to create a non-linear network and fits complex data. It is defined as : relu(x) = { 0 if x<0, x if x > 0}. In the definition of the ReLU function, it renders a positive number as the number itself, while for negative number, it returns 0. After applying convolution and ReLU, we use a pooling layer. We use also MaxPool2D, which has shown much better performance and is generally the preferred pooling strategy for large-scale computer vision tasks.

Finally, the output in our code is passed through the dropout layer, which is a random mask of the output, equivalent to randomly centering the input to the next layer during training time with probability. During the test, the dropout layer is removed and all weights are used. This prevents overfitting and acts as a regulator for the neural network, although the best value for probability has to be found experimentally. In our model design, we implemented the CNN module several times for the deep multilayer framework.

3.3. Graph Convolutional Networks

Convolutional neural networks can solve problems with ordinary 1-D and 2-D Euclidean data such as image and text classification, but often real-world data has a non-Euclidean structure. In this stage, graph neural networks become a solution that allows us to capture rich features of complex relationships between data. In recent years, various variants of graph neural networks are being developed with graph convolutional networks (GCN) being one of them.

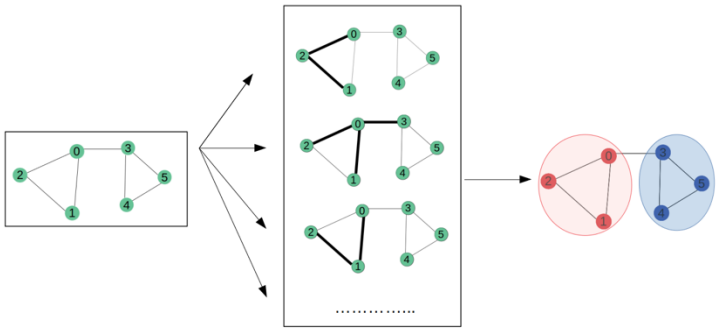


Figure 6: Illustration of graph convolutional networks (GCN).

The term 'convolution' in graph convolutional networks is similar to convolutional neural networks in terms of weight distribution. GCN performs a similar operation as CNN (by multiplying the input neurons with a set of weights), where the model learns features by examining neighboring nodes. The main difference lies in the data structure, where GCN is a generic version of CNN that can work on data with an underlying non-regular structure. The general idea of GCN is to apply convolution to the graph by taking graph as an input, instead of having a 2-D array as input.

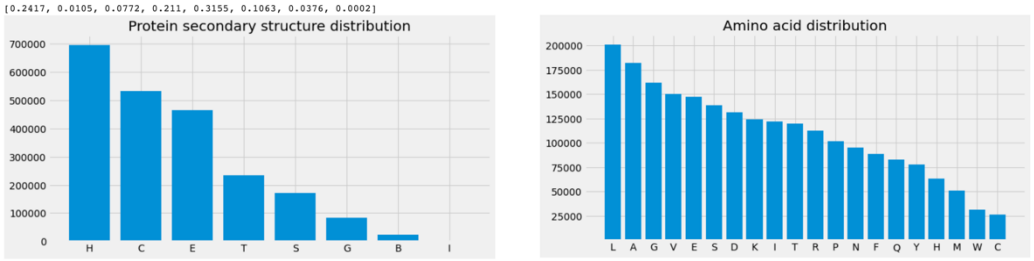
For our model, we start the work by doing some data processing. Based on the given dataset, the column 'seq' has the primary protein sequence and 'sst8' has the secondary protein sequence that we will predict. The max length of each sequence is set to 128. The wording of hasnonstdaa means the peptide contains nonstandard amino acids (B, O, U, X, or Z). So the sequence is only taken that does not have nonstandard amino acids. Then our model with GSN will look for incomplete data by checking whether there are differences and the number of characters of the primary and secondary structures. This is because the secondary structure prediction is included in the sequence labeling problem, so it can be ascertained that the number of primary and secondary structure characters is always the same.

Next time we do is orthogonal coding and target labeling. Orthogonal coding is used to extend ensemble coding because it includes the notion that it is either on or off. It means that each primary and secondary data structure is separated so that it can be encoded into an orthogonal form. Then we do the split function by splitting the string sequence into character array.

4 Results

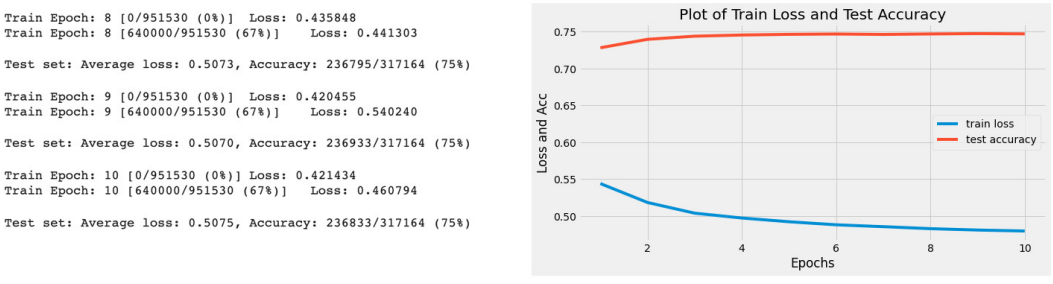
4.1. Convolutional Neural Networks

For the data preparation, we have divided the dataset to be 75% for training and 25% for testing purpose. Once we have imported libraries and read the data uploaded, we plot the figures of secondary structure and amino acid distribution (figure 7).



(a) (b)

Figure 7: Plot of (a) Protein secondary structure distribution, (b) Amino acid distribution.



(a) (b)

Figure 8: (a) Result of running train epoch, (b) Plot of train loss vs. test accuracy.

For the convolutional neural network with sliding windows, the prediction accuracy and plot of train loss vs. test accuracy are shown in figure 8. The accuracy on the test set achieved with this model is equal to **75%** based on the given dataset for the Q8 prediction problem.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 32, 19, 18]             320
       BatchNorm2d-2          [-1, 32, 19, 18]              64
              ReLU-3          [-1, 32, 19, 18]               0
            Conv2d-4          [-1, 64, 17, 16]          18,496
       BatchNorm2d-5          [-1, 64, 17, 16]             128
              ReLU-6          [-1, 64, 17, 16]               0
         MaxPool2d-7            [-1, 64, 8, 8]               0
           Dropout-8            [-1, 64, 8, 8]               0
            Linear-9                 [-1, 128]         524,416
      BatchNorm1d-10                 [-1, 128]             256
             ReLU-11                 [-1, 128]               0
           Linear-12                  [-1, 64]           8,256
      BatchNorm1d-13                  [-1, 64]             128
             ReLU-14                  [-1, 64]               0
          Dropout-15                  [-1, 64]               0
           Linear-16                   [-1, 3]             195
================================================================
Total params: 552,259
Trainable params: 552,259
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.72
Params size (MB): 2.11
Estimated Total Size (MB): 2.82
----------------------------------------------------------------
```

Figure 9: The summary of our model using window CNN.

4.2. Graph Convolutional Networks

The results of the separation of primary and secondary protein structures are then converted into orthogonal encoding and target labeling. Secondary structure encode the 8 classification using numbers 0-7 (see figure 10).

```
'H' : 0,     # H= α-helix
'C' : 1,     # C= Loops or irregular
'E' : 2,     # E= β-strand
'B' : 3,     # B= β-bridge
'G' : 4,     # G= 3-helix
'I' : 5,     # I= π-helix
'T' : 6,     # T= Turn
'S' : 7      # S= Bend
```

Figure 10: Representation of secondary structure character.

The primary structure consists of a linear sequence of amino acids (nodes), for example: ABBA. In graph neural network, they own each node and add up all the information values of the adjacent nodes. As a result, it assigns a new value for each node which is completely reliable for adjacent nodes. Then we create a secondary structure in the data array using the targetY function. The data feature is created using the window_padding_data function, which receives the size of the sliding window and the sequence of primary structure. In this function, the features that will be processed are taking features from windowing results so that the output data can be directly trained on the SVM model.

Before starting the Scikit-Learn SVM model, the data is reshaped to fit the size of the model input. The data is remodeled to length X (main structure) * 220 (x window size and 20 orthogonal coding sizes.). Finally, the data is divided into training and testing data to get the final result. Then it is

calculated by SVM and visualized with the classification report. The accuracy rate of the model constantly stays around **~60%** (see figure 11).

Accuracy = 60.45694200351493

	precision	recall	f1-score	support
0	0.68	0.86	0.76	296
1	0.50	0.90	0.64	345
2	0.86	0.50	0.63	241
3	0.00	0.00	0.00	8
4	0.00	0.00	0.00	34
5	0.00	0.00	0.00	2
6	1.00	0.01	0.02	105
7	1.00	0.01	0.02	107
accuracy			0.60	1138
macro avg	0.50	0.28	0.26	1138
weighted avg	0.70	0.60	0.53	1138

Figure 11: Accuracy result of our model using GCN.

5 Conclusions

In conclusion, the developed models were able of replicating the state-of-art results with great efficiency for 8-state prediction (Q8). From the above results we can see what we can get the best performance using convolutional neural networks (CNN) with sliding window (with accuracy equal to 75%). This model only focuses on modelling local structure to predict the secondary structure label, which is the opposite to the state-of-art model which models local and global structures. although the results we obtained from the GSN model are not too bad (accuracy equal to ~60%), but it turns out that modelling with CNN can produce higher accuracy. For a future project, we believe that extending our model to include a global structure can yield a better result.

In the end our main takeaway from this project is that local structure modelling is the most important aspect to get good accuracy and adding global structure will be useful to increase that accuracy by some amount.

References

[1] Noble M.E., Jane E., and Louise N.J. (2004) Protein kinase inhibitors: insights into drug design from structure. *Science*, 303(5665):1800– 1805.

[2] Pauling, L., Corey, R. B. & Branson, H. R. (1951) The structure of proteins: two hydrogen-bonded helical configurations of the of the polypeptide chain. *Proc. Natl. Acad. Sci*. USA 37, 205–211.

[3] Kabsch W. and Sander C. (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers.* PubMed.

[4] Qian, N. and Sejnowski, T. J. (1988)Predicting the secondary structure of globular proteins using neural net- work models. *Journal of molecular biology*, 202(4):865– 884, 1988. ISSN 0022-2836.

[5] Holley L. H. & Karplus M. (1989) Protein secondary structure prediction with a neural network. Proc. Natl. Acad. Sci. USA 86, 152–156. PubMed.

[6] Rost, B. and Sander, C. (1993) Prediction of protein secondary structure at better than 70accuracy. *Journal of molecular biology*, 232(2):584–599, 1993. ISSN 0022- 2836.

[7] Altschul S.F., Madden T.L., Schäffer A.A, Zhang, J. Zhang Z., Miller W. and Lipman D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res. 25*, 3389–3402. Gogle Scholar.

[8] Baldi, P., Brunak, S., Frasconi, P., Soda, G., and Pollastri, G (1999). Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, 15(11):937–946, 1999. ISSN 1367- 4803.

[9] Schmidler, S.C., Liu, J.S., and Brutlag, D.L. (2000) Bayesian segmentation of protein secondary structure. *Journal of computational biology*, 7(1-2):233– 248, 2000. ISSN 1066-5277.

[10] van der Maaten, L., Welling, M., and Saul, L.K. (2011) Hidden-unit conditional random fields. *Journal of Machine Learning Research-Proceedings Track*, 15:479–488, 2011.

[11] Heffernan R., Yang Y., Paliwal K., Zhou Y. (2017) 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*. Google Scholar.

[12] Szegedy C., Vanhoucke V., Ioffe S., Shlens J. and Wojna Z. (2016) Rethinking the Inception Architecture for Computer Vision. *Conference on Computer Vision and Pattern Recognition (CVPR*). Los Alamitos: IEEE Computer Society.

[13] Jones, D. T. (1999) Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology* 292(2):195–202.

[14] Drozdetskiy, A., Cole, C., Procter, J. and Barton, G. J. (2015) JPred4: a protein secondary structure prediction server. *Nucleic Acids Research* gkv332.

[15] Qi, Y., Oja, M., Weston, J. and Noble, W. S. (2012) A unified multitask architecture for predicting local protein properties. *PloS one* 7(3):e32235.

[16] Zhou J. and Troyanskaya O.G. (2014) Deep supervised and convolutional generative stochastic network for protein secondary structure prediction. *Proceedings of the 31st International Converenfe on Machine Learning (ICML)*. Bejing: PMLR.

[17] Darnell S. (2020) Why structure prediction matters. In *Structural Biology.* DNA Star.

[18] Cuff J.A. and Barton G.J. (2000) Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Wiley Online Library*.

[19] Afifi H.M., Abdelhalim M.B., Mabrouk M.S., and Sayed A.Y. (2021) Protein secondary structure prediction (PSPP) using different machine learning algorithms. *Egyptian Journal of Medical Human Genetics*, volume 22. Springer.

[20] Suh D., Lee J.W., Choi S. and Lee Y. (2021) Recent applications of deep learning methods in evolution- and contact- based protein structure prediction. *International Journal of Molecular Science*. National Library of Medicine.

[21] Ratul M.A.R, Turcotte M., Mozaffari M.H. and Lee W. (2019) Prediction of 8-state protein secondary structures by 1D-inception and BD-LSTM. *Publications of Research*. ResearchGate.

[22] Cheng J., Liu Y. and Ma Y. (2020) Protein secondary structure prediction based on integration CNN and LSTM. *Journal of Visual Communication ad Image Representation,* volume 71. Elsevier

[23] Zhang B., Li J. and Lu Q. (2018) Prediction of 8-state secondary structures by a novel deep learning architecture. *Article of BMC Informatics*, volume 19. National Center for Biotechnology Information.

[24] Pakhrin S. C., Shrestha B., Adhikari B. and Kc D.B. (2021) Deep learning-based advances in protein structure prediction. *International Journal of Molecular Science*. National Library of Medicine.

[25] Yang Y.D., Gao J.Z., Wang J.H., Heffernan R., Hanson J., Paliwal K., and Zhou Y.Q. (2018) Sixty-five years of the long march in protein secondary structure prediction: The final stretch? *Briefings Bioinf.*, vol. 19, no. 3.

[26] Agrawal et al., (2019) ccPDB 2.0: an updated version of datasets created and compiled from Protein Data Bank. https://webs.iiitd.edu.in/raghava/ccpdb/

[27] Agrawal et al., (2019) ccPDB 2.0: an updated version of datasets created and compiled from Protein Data Bank. https://webs.iiitd.edu.in/raghava/ccpdb/datasets/dssp-dataset-normal.txt