Machine Learning-Based Determination of Protein Secondary Structures

Huan-Jui Lee

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Chemical Engineering

University of Washington

2021

Reading Committee:

René M. Overney, Chair

David A.C. Beck

Program Authorized to Offer Degree:

Chemical Engineering

University of Washington

Abstract

Machine Learning-Based Determination of Protein Secondary Structures

Huan-Jui Lee

Chair of the Supervisory Committee:
Professor René M. Overney
Department of Chemical Engineering

The main purpose of this thesis is to construct a machine learning model that yields protein secondary structures from sequences and circular dichroism (CD) spectra, and test the contribution of each part. This effort is motivated by the desire to reduce the costs and time involved in state-of-the-art approaches, which involve elaborate instrumentation, such as nuclear magnetic resonance (NMR) and X-ray powder diffraction (XRD). Conformational analysis based on current experimental methods require preparations and analytical processes that are often hampered by sample impurities and aging, and, limitations originating from crystal cultures. A well-developed machine learning algorithm, based on existing conformational data provides an easier and also faster way to predict unknown conformations of proteins. In the research here, we make use of CD spectra and improvement of machine learning model. The algorithm used in this thesis is based on Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN), we analyzed the performance of single model and stacked model. The result indicates that stacked model and CD spectra can help us to improve the accuracy of prediction.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor, Professor René M. Overney, for his encouragement and guidance, Professor David A.C. Beck for being my committee member, my group members, Tyler Jorgenson, YouHsin Chen, Saransh Jain, Harini Kethar, Eric Pattison, for their support and ideas, and my parents for their unconditional support and inspiration.

Chapter 1. INTRODUCTION

As the most important biological molecule in cells, protein exhibits a wide variety of functions. Its functions are directly related to its properties and interactions, which in turn can be inferred from analyzing its structure. [1,2] With knowledge and control over protein functions, proteins can by targeted towards drug development and biological material production[3,4]. Therefore, the evolution of key technologies for obtaining reliable and swift protein structure information has been on-going.

In general, there are four levels of protein structures: (i) the primary structure, (ii) the secondary structure, (iii) the tertiary structure and (iv) the quaternary structure. In this research, our focus is on obtaining secondary structures from primary structure information. Primary structures represent the amino-acid sequences that make the protein. The secondary structure captures the local folded structures that form within a polypeptide due to interactions between the atoms of the backbone[5]. The most common types of secondary structures are the α helix and the β pleated sheet. Responsible for these two structures are hydrogen bonds that form between the carbonyl O of one amino acid and the amino H of another. Propensity of secondary structure is affected by both short and long interactions. Each amino acid has its own propensity, but adjacent amino acids are also factors to determine the secondary structure[6].

The most popular experimental methods developed to obtain secondary structures are nuclear magnetic resonance (NMR)[7,8] and X-ray powder diffraction (XRD)[9]. NMR and XRD analyze the different conformations of peptides by mapping out the different associated energy levels, which is known as conformational analysis. Conformational analysis based methods yields high precision and reliability information. However, the long execution time, demand for sophisticated instruments and requirement of purity of samples make these methods time-consuming and of high costs.

In order to reduce the cost, machine learning methods have been utilized in the prediction of protein secondary structure in recent decades[10,11]. A well-developed machine learning algorithm based on existing conformational data provides an easier and also faster way to predict secondary structure of unknown proteins. Instead of implementing complicated experimental analysis of individual samples, trained machine learning model only needs essential parameters to get the structure. In this thesis, we applied Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN) and stacked technique.

To achieve higher prediction accuracy, we made use of Circular dichroism (CD) to assist our result. Circular dichroism evaluates the composition of secondary structures, which means the percentage of each secondary structure in the target protein, through spectroscopy. It measures the absorption spectra difference of protein between left and

right-handed circularly-polarized light[12,13]. Since the structure of protein affects its absorbed circular polarized light and lead to distinct CD spectra, we could determine secondary structure composition by its CD spectra pattern. We chose CD analysis, because it is quickly obtained (in less than one hour) and involves comparable to NMR and XRD inexpensive equipment. Ultimately the objective of this research is that researchers can obtain from protein sequences and CD spectra with our machine learning model fast predictions of secondary protein structures.

Chapter 2. EXPERIMENTAL METHODS

2.1 Data

The dataset used in this thesis is from the PISCES server[14], which provide culling sets of proteins from PDB by percentage identity cutoff of sequences, resolution and R-factor. The percentage identity cutoff represents the similarity of the proteins in the dataset; R-factor measures the degree of matching between simulated diffraction pattern and the experimentally-observed diffraction pattern. In order to make model be more generalized and has extensive application, we chose cullpdb_pc20_res1.6_R0.25_d201015 with only 20% identity. The cullpdb contains about 3,600 proteins, some of them were removed due to error occurred in sequence alignment. The sequences and secondary structures of remaining 3594 proteins were extracted via BioPython, which is a Python package designed for biological usage. Secondary structures were classified into 8 classes by Dictionary of Secondary Structure of Proteins (DSSP)[15], as shown in Table 1.

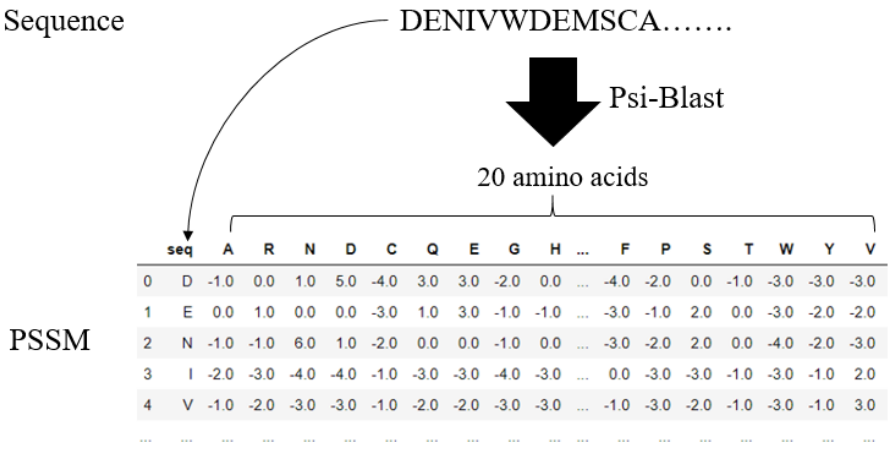
Table 1. 8-classes DSSP code and corresponding structure

Code	Structure
H	alpha helix
B	residue in isolated beta-bridge
E	extended strand
G	3-helix (3/10 helix)
I	5 helix (pi helix)
T	hydrogen bonded turn
S	bend
-	other

2.2 Sequence Alignment

As a list of letters, sequence need to be converted to value, vector or matrix before importing into machine learning model. In addition, sequence is barely one dimensional and lacks of information, so it is not suitable training data before further processing. For the sake of improving the performance of our machine learning model, sequence alignment was used to get more information in our dataset and increase the number of features. Sequence alignment is a widely utilized method in data processing of protein or DNA[16,17]. It searches and identifies similar parts of sequences in other proteins within its database, and shows log-likelihoods of the occurrence probabilities of all amino acids in that position. Therefore, the dimension of output is same as number of kinds of amino

acids in sequences. We used Psi-Blast[18] to do sequence alignment in this research, and selected PDB protein database (pdbaa) as reference database. A 20-dimensional position specific scoring matrix (PSSM) created after three iterations searching for every single protein, as shown in Figure 1. Figure 1 reveals the appearance of PSSM which contains the pattern of each specific group, and provides the variance of amino acid positions within a specific protein group based on relevance[19].



	seq	A	R	N	D	C	Q	E	G	H	...	F	P	S	T	W	Y	V
0	D	-1.0	0.0	1.0	5.0	-4.0	3.0	3.0	-2.0	0.0	...	-4.0	-2.0	0.0	-1.0	-3.0	-3.0	-3.0
1	E	0.0	1.0	0.0	0.0	-3.0	1.0	3.0	-1.0	-1.0	...	-3.0	-1.0	2.0	0.0	-3.0	-2.0	-2.0
2	N	-1.0	-1.0	6.0	1.0	-2.0	0.0	0.0	-1.0	0.0	...	-3.0	-2.0	2.0	0.0	-4.0	-2.0	-3.0
3	I	-2.0	-3.0	-4.0	-4.0	-1.0	-3.0	-3.0	-4.0	-3.0	...	0.0	-3.0	-3.0	-1.0	-3.0	-1.0	2.0
4	V	-1.0	-2.0	-3.0	-3.0	-1.0	-2.0	-2.0	-3.0	-3.0	...	-1.0	-3.0	-2.0	-1.0	-3.0	-1.0	3.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 1. Generating PSSM via Psi-Blast

2.3 Encoding Method of Secondary Structure

Same as sequence, the secondary structure of proteins is also a list of letters, so it need to be encoded as value/vector like sequence alignment did. Table 2 lists common encoding methods for proteins[20]. Here we chose one-hot. In a “one-hot” format, where categorical variables are represented by binary vectors, with a single high (1) bit which indicates the

secondary structure of the corresponding amino acid and all the others low (0), as shown in Figure 2.

Table 2. Common encoding method of proteins

Category	Encoding Method
Binary	One-hot
	One-hot (6-bit)
	Binary 5-bit
Physiochemical properties	Hydrophobicity matrix
	Meiler parameters
	Acthely factors
Evolution-based	PAM250
	BLOSUM62
	PSSM
	HMM
Structure-based	Miyazawa energies
	Micheletti potentials
Machine-learning	AESNN3
	ANN4D
	ProtVec
	ProVec-3mer

H	B	E	G	I	T	S	-
0	0	0	0	0	0	0	1
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...
0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1

Figure 2. Label data in one-hot format

2.4 First Machine Learning Model

The first machine learning model predicts secondary structures by sequences. A two-level stacked model constructed with Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) as shown in Figure 3. Level-0 contains three independent models using LSTM, CNN and combination of these two algorithm. The training data were separated equally and imported into the models. Level-1 is the model which combined outputs from three level-1 models.

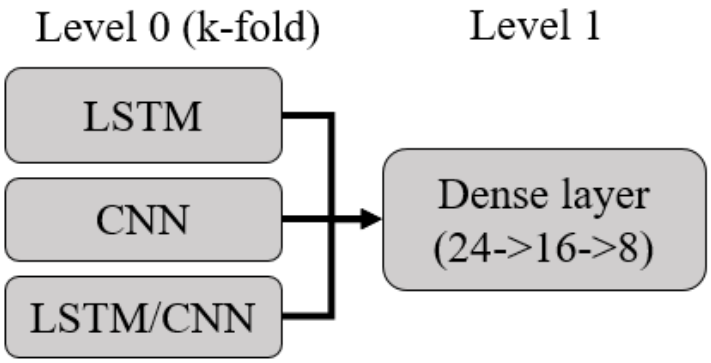


Figure 3. Two-level stacked model

2.4.1 Long Short-Term Memory (LSTM)

A variety of machine learning methods have been implemented to determine protein secondary structures. Many of common algorithms are non-sequential, which means the influence from previous and next input is not included[21,22]. However, the propensities of secondary structures are not only affected by corresponding sequence (amino acid) but also influenced by the adjacent amino acids. The main reason we chose LSTM because it is well suited for data with order like sequences or timeline[23,24]. Figure 4 illustrates the mechanism of LSTM. LSTM adds weight got previously to next block, so it conforms the feature that adjacent amino acids would affect the secondary structure of the target amino acid. Figure 5 reveals our LSTM model that was composed of three LSTM layers, one timedistributed layer to dense the output from previous layer in every time step, and two dense layer in final. We also added Gaussian Noise layer and dropout layer to reduce

overfitting[25,26]. In order to match the format of our secondary structures (one-hot encoding), we chose “softmax” as the activation function of the output layer, and categorical crossentropy as the loss function in the model. The output of Softmax function is probabilities of all categories with sum to 1. Categorical crossentropy is the loss function that matches one-hot encoding.

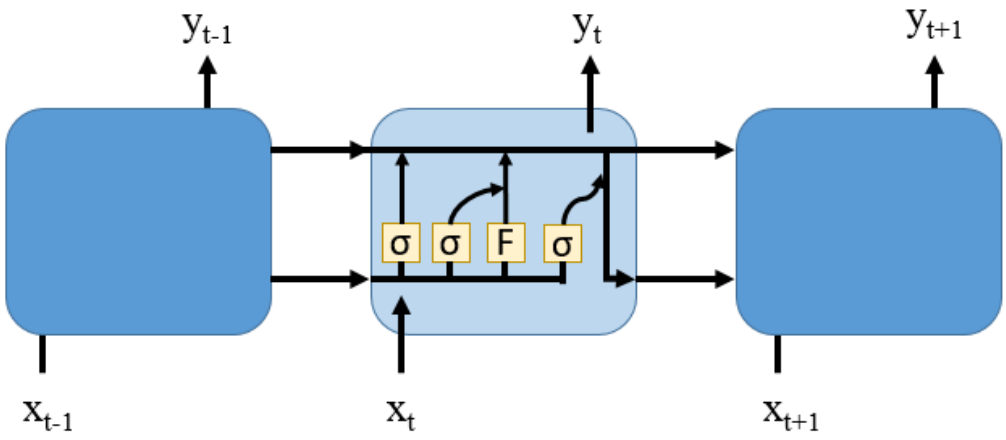


Figure 4. Mechanism of LSTM model

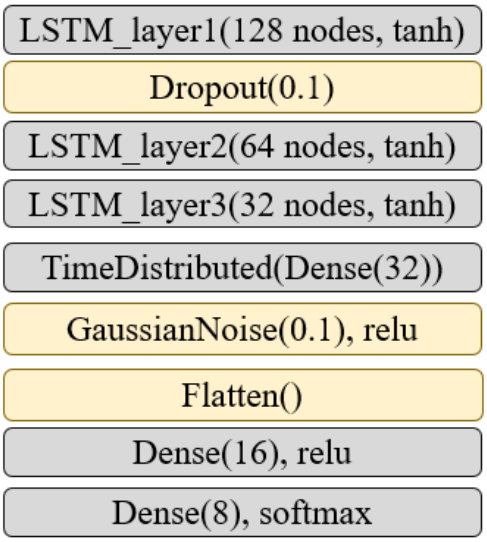


Figure 5. Composition of the LSTM model

2.4.2 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a popular method in prediction of protein secondary structure[27,28]. In general, a CNN model consists of convolutional layer, pooling layer, and fully-connected (dense) layer. Convolutional layer would grab local features in dataset. Pooling layer reduces the dimension of training data and keep stronger features to mitigate computational load and avoid overfitting. Fully-connected layer connects output. Our CNN model contains four convolutional layers, two pooling layers and three dense layers as shown in Figure 6. The activation function of output layer and loss function are same as LSTM model.

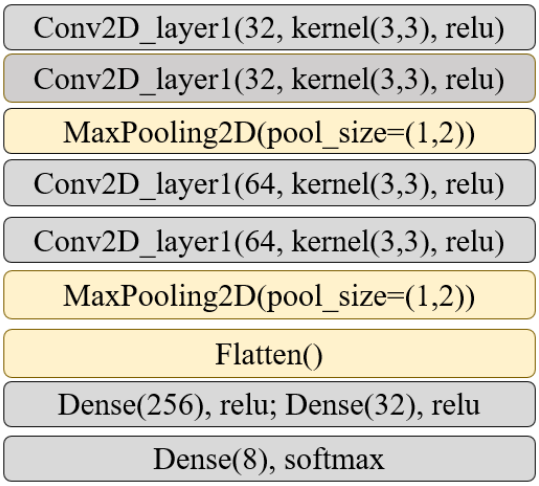


Figure 6. Composition of the CNN model

2.4.3 Stacked Model

Combining the output results of different models within their own advantages can enhance the performance, and stacked model is a good way to achieve it[29,30]. Figure 7 illustrates how our stacked model works. The model has a dense layer to concatenate the output of three level-0 models from (none, 8) to (none, 24), and the next two dense layers would keep training to find the best way to combine their results.

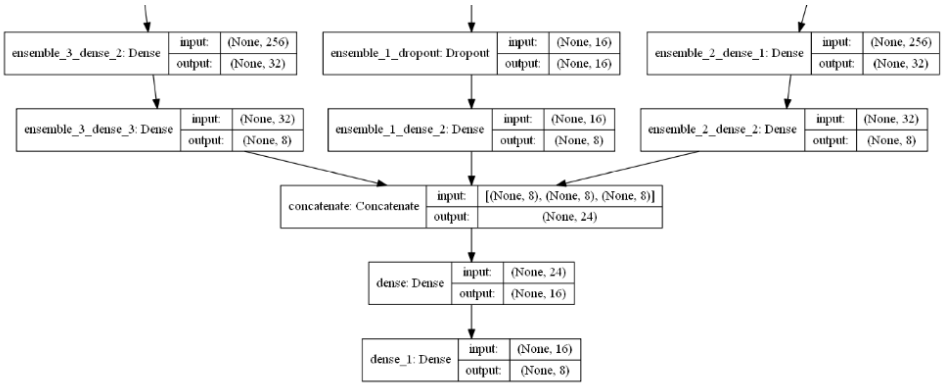


Figure 7. Composition of stacked model

2.4.4 Customize Weighted Loss Function

Compare to three-classes secondary structure, eight-classes would face imbalanced data problem. Table 3 shows the percentage of secondary structures of our dataset. Alpha helix (H), extended strand (E) and other (-) account for 75% of data, and other five classes of secondary structure only account for 25%. Machine learning model would tend to classify all data to these three categories when implementing training with imbalanced dataset straightly, because it can still get 75% accuracy. However, this phenomenon lets our model loss the ability to detect other five classes secondary structure. Even if the accuracy was high, the model has no practical application value.

Table 3. Distribution of secondary structures in dataset

Structure	Amount	Percentage
H	406383	31.8%
B	15055	1.2%
E	294141	23.0%
G	51452	4.0%
I	6727	0.5%
T	143943	11.2%
S	105928	8.3%
-	256290	20.0%

Imbalanced dataset is a common issue in machine learning, researchers utilize various methods to solve this problem[31]. In this thesis, we customized weights of each secondary structure in loss function based on their distribution in data. We reduced weights of three

main classes and increased of which other five classes. Adjusted amounts depended on the percentage of original data.

2.5 Second Machine Learning Model and Output Processing

The second model is to evaluate proportion of three-classes secondary structures in each protein by CD spectra. We utilized the result of this model to assist the determination of secondary structure of our first model. The algorithm of the second machine learning model is based on CNN, and CD spectra data were generated by Structure-Based Empirical Spectrum Calculation Algorithm (SESCA)[32,33] within a wavelength regime of 178-260 nm, as shown in Figure 8, using the same protein data as the first model and extracting secondary structures data in PDB files for calculation. Since SESCA extracted lots of information from PDB file to get secondary structure coefficients and calculated CD spectra, the second model has more information compares to the first model. Simultaneously, one set of CD spectra was used to predict only three parameters: proportion of alpha helix, beta sheet and random coil. Therefore, the accuracy of the second model would be much higher than the first model, which make it could be utilized as correction tool to improve result from the first model.

We first compared the proportion of three-states secondary structures getting from two models. If the total differences of percentages excess 20%, all categories that are higher

than the results of the second model by more than 5% would be reclassified. Since Softmax was the activation function chose in our first model, the original output had eight secondary structure probabilities for each amino acids and the highest one was picked. When the output satisfied two conditions that it belonged to the category that needed to be corrected, and the prediction probability of that category was lower than a threshold value, we moved output from highest value to second highest, third highest, fourth highest and fifth highest values and chose the one had best final accuracy. Threshold value was determined by value of percentile 35, 25, 15 of output that classified as this class. As a result, for the case that one category needed to be reclassified, we tested 12 combinations and selected the set with best accuracy as the final output. In addition, there is a minimum length rule for specific secondary structure in DSSP format, as shown in Table 4. We also used this rule to revise our output after correction from the second model.

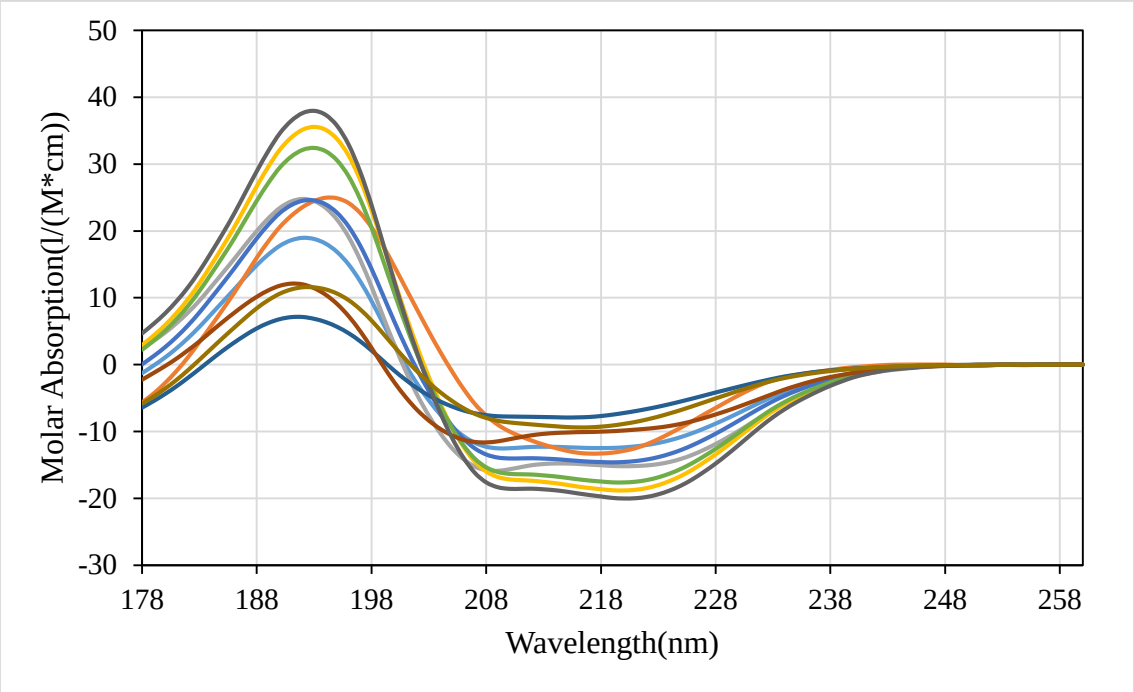


Figure 8. Calculated CD spectra generated by SESCA

Table 4. Minimum length N residues rule of DSSP

Structure	Minimum length N residues
H	4
G	3
I	5
T	3,4 or 5 turn
E	2

Chapter 3. RESULTS AND DISCUSSION

3.1 Effect of Customized Weighted Loss Function

Basically, the accuracy of model remained as the same level of original result after customizing the weights, so the efficiency of reweighting is implicit that could not be detected directly. We evaluated the efficiency of customized weighted loss function via confusion matrix in Figure 9.

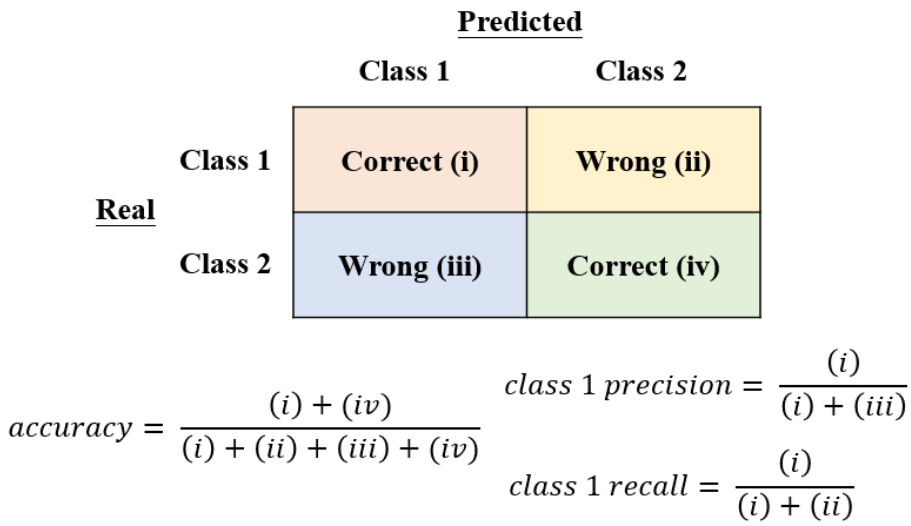


Figure 9. Confusion matrix analysis

In addition to listing whole confusion matrix and investigate prediction accuracy for all categories, we used two indicators mentioned in Figure 9 to describe our model's performance. Precision represents how precise when the model determines class of the target, so we could figure out the correctness for this predicted class. Recall illustrates the ability of the model to find the true label and classify correctly. The intention of adjusting

the weights of loss function is to improve the performance of our model on detecting minor groups, so recall is the indicator that we focused on. Table 5 lists precision and recall of eight secondary structure classes under six sets of weights in LSTM model.

Table 5. Precisions and recalls of output under different weights by LSTM model

Weights	Indicator	H	B	E	G	I	T	S	-
[1., 1., 1., 1.,	precision	56%	0%	49%	50%	84%	24%	22%	31%
1., 1., 1., 1.]	recall	**76%**	**0%**	**60%**	**8%**	**2%**	**13%**	**2%**	**32%**
[0.3, 1., 0.7, 1.,	precision	67%	0%	42%	33%	68%	22%	18%	31%
1., 1., 1., 0.5]	recall	**48%**	**0%**	**71%**	**23%**	**20%**	**34%**	**17%**	**17%**
[0.5, 1., 0.7, 1.,	precision	65%	0%	48%	40%	75%	25%	22%	30%
1., 1., 1., 0.7]	recall	**58%**	**0%**	**63%**	**24%**	**17%**	**28%**	**11%**	**33%**
[0.6, 1.3, 0.8, 1.,	precision	62%	0%	51%	43%	80%	25%	21%	30%
1., 1., 1.2, 0.8]	recall	**65%**	**0%**	**60%**	**18%**	**9%**	**24%**	**14%**	**33%**
[0.6, 1., 0.9, 1.1,	precision	62%	0%	48%	37%	76%	24%	22%	31%
1.1, 1.1, 1.1, 0.8]	recall	**64%**	**0%**	**65%**	**23%**	**12%**	**26%**	**9%**	**28%**
[0.6, 1., 0.8, 1.1,	precision	62%	0%	49%	45%	74%	26%	21%	30%
1.2, 1.1, 1.2, 0.8]	recall	**64%**	**0%**	**61%**	**21%**	**22%**	**23%**	**14%**	**31%**

The results show that when we reduce the weights of the main components, the recalls of the other five secondary structures increase significantly. However, the recalls of main classes have decreased, which is basically an inevitable compensation effect. Therefore, recklessly reducing the weight of the main classes, like the second set of weights in Table 5, is negative to the final predictive ability of the model, and may even affect the accuracy. The moderate adjustment of weights should be based on the real distribution in data, and minimize the compensation reductions of main classes. The final weights we selected and

entered customized loss function is the sixth set, which only resulted in deduction of alpha helix's recall but contributed manifest improvement of recalls for four secondary structures.

The recall of isolated beta-bridge residue (B) is the only one that did not improve. The first reason is that it barely accounted for 1.5% in our data, so the promotion of recall is not obvious. The second reason is related to the original forecast distribution. Table 6 and Table 7 are confusion matrices with no customized weights and final customized weights. Comparison of two tables points out that the four minor classes got improvement due to the decreasing of probabilities to be predicted as alpha helix inaccurately. They were also mostly classified to alpha helix in original model. On the other hand, isolated beta-bridge residue was mostly classified to other (41%), of which probability was much higher than proportion of alpha helix (25%) or extended strand (24%). Thus, adjusting weights is not beneficial to recall of its classification. Even the model using the second sets of weights with lower value of other did not give rise to visible difference. The third reason is that alpha helix was not re-classified as isolated beta-bridge residue, which caused by higher proportion of hydrogen bonded turn and bend in dataset.

Table 6. Confusion matrix of LSTM model output without customized weights

Real \ Predicted	H	B	E	G	I	T	S	-
H	76%	0%	12%	0%	0%	2%	0%	10%
B	25%	0%	24%	1%	0%	8%	2%	41%
E	19%	0%	61%	0%	0%	3%	1%	17%
G	38%	0%	16%	8%	0%	9%	1%	29%
I	73%	0%	19%	0%	3%	1%	0%	4%
T	43%	0%	19%	1%	0%	13%	1%	24%
S	29%	0%	23%	0%	0%	11%	3%	34%
-	29%	0%	27%	0%	0%	10%	2%	32%

Table 7. Confusion matrix of LSTM model output with best customized weights

Real \ Predicted	H	B	E	G	I	T	S	-
H	64%	0%	15%	1%	0%	5%	2%	12%
B	15%	0%	22%	2%	0%	13%	10%	39%
E	12%	0%	61%	0%	0%	5%	4%	18%
G	22%	0%	14%	21%	0%	12%	4%	27%
I	53%	0%	17%	0%	22%	3%	1%	4%
T	30%	0%	18%	2%	0%	23%	5%	23%
S	18%	0%	21%	2%	0%	16%	14%	30%
-	19%	0%	26%	1%	0%	14%	9%	31%

3.2 Results of Stacked model

Table 8 shows the performance of our model. LSTM and CNN model in level-0 are almost same in predictive ability. LSTM+CNN model has better validation accuracy, but the difference is not apparent. The stacked model enhanced prediction ability effectively from average accuracy 0.5360 and validation accuracy 0.4664 in level-0 models to accuracy 0.5826 and validation accuracy 0.4923. It proves that stacked model could definitely combine multiple member input models and optimize the performance. In order to further analyze the impacts of stacked model, we recorded stacked models with different combination of level-0 models in Table 9. The model with LSTM and CNN slightly increases the validation accuracy, but the improvement is not obvious and worse than individual model using CNN+LSTM. The other two combinations have distinct improvement in both accuracy and validation accuracy, especially the latter is close to best performance. However, all two-models combinations cannot provide better results than completed stacked model.

Table 8. Performance of the first machine learning model

Level	Program	Validation split	loss	accuracy	Val_ loss	Val_ accuracy
0	LSTM	10%	1.0935	0.5463	1.3097	0.4654
	CNN	30%	1.0919	0.5343	1.2847	0.4617
	CNN+ LSTM	30%	1.1015	0.5276	1.2468	0.4722
1	Stacked model	30%	1.1523	**0.5826**	1.4368	**0.4923**

Table 9. Performance of stacked models with different combination

Program	accuracy	Val_accuracy
Single LSTM	0.5463	0.4654
Single CNN	0.5343	0.4617
Single CNN+LSTM	0.5276	0.4722
LSTM & CNN	0.5066	0.4711
LSTM & CNN+LSTM	0.5737	0.4900
CNN & CNN+LSTM	0.5520	0.4920
LSTM & CNN& CNN+LSTM	0.5826	0.4923

3.3 Results After Output Processing

The feasibility of enhancing our model by output processing is partly based on the accuracy of our second machine learning model, which is shown in Table 10. According to three-classes label data and high-reliability calculated CD spectra generated by SESCA, we got high accuracy in both training and validation data. The good performance was beneficial to our output processing.

Table 10. Performance of the second machine learning model

Program	Validation split	loss	accuracy	Val_loss	Val_accuracy
CNN	30%	0.9124	**0.9700**	0.9231	**0.8916**

During output processing, the accuracy of individual protein could be improved by 0~8%, depending on its original classification distribution. The improvements by reclassifying alpha helix were more stable, about 2.5%. The enhancements from reclassification of beta sheet and random coil were relatively drastic, from almost 0 to an increase of 8%. Table 11 shows the final accuracy of our model after assisting by second machine learning model and minimum length rule. Both accuracy and validation accuracy increase about 2.5%, indicating that method using CD spectra and DSSP rule is rewarding. We did confusion matrix analysis to our final result and compared with LSTM model in Table 12 and Table 13 to see the improvement from single machine learning model to stacked

model with correction from CD spectra and minimum length rule. Our final model performs better than LSTM model in nearly all classes' precision and recall.

Table 11. Comparison of performance before and after output processing

Program	loss	accuracy	Val_loss	Val_accuracy
Stacked model	1.1523	0.5826	1.4368	0.4923
After output processing	--	0.6074	--	0.5179

Table 12. Confusion matrix of the final model

Real \ Predicted	H	B	E	G	I	T	S	-
H	70%	0%	11%	1%	0%	4%	2%	12%
B	17%	0%	19%	2%	0%	12%	9%	42%
E	14%	0%	60%	1%	0%	4%	3%	18%
G	24%	0%	11%	27%	0%	8%	3%	26%
I	53%	0%	9%	0%	33%	2%	0%	3%
T	32%	0%	15%	2%	0%	24%	4%	23%
S	20%	0%	19%	2%	0%	14%	13%	32%
-	21%	1%	23%	2%	0%	13%	8%	33%

Table 13. Comparison of performance between LSTM model and the final model

Model	Indicator	H	B	E	G	I	T	S	-
LSTM model	precision	62%	0%	49%	45%	74%	26%	21%	30%
	recall	**64%**	**0%**	**61%**	**21%**	**22%**	**23%**	**14%**	**31%**
Final model	precision	62%	2%	53%	47%	71%	30%	24%	31%
	recall	**70%**	**0%**	**60%**	**27%**	**33%**	**24%**	**13%**	**33%**

Chapter 4. CONCLUSION

This study has determined the effectiveness of the stacked model and CD spectra in improving the accuracy of protein secondary structure prediction. We examined the performance of model using different machine learning methods to determine protein secondary structures from sequences. The algorithms we utilized involve LSTM, CNN, LSTM+CNN, stacked models which combined any two of three individual models, and stacked model that concatenated all three models. The results indicate that stacked model that integrated three models has best performance, which is 2.5% over the average of single models, and stacked model containing two models follow. Meanwhile, we studied the feasibility of improving our classification by minimum length rule in DSSP and the second machine learning model that predicts proportion of secondary structure from CD spectra generated by SESCA. Both accuracy and validation accuracy of the second machine learning model are favorable. The performance we obtained shows that they are indeed beneficial to enhance the accuracy, up to also about 2.5%. Our final accuracy in 3-states secondary structure is at the same level (about 0.89) as other studies[30]. Accuracy in 8-states (0.518) is lower than other similar studies[21,34], which could achieve 0.654 and 0.674. The difference may be caused by different training datasets, algorithm and data preprocessing, which needs further confirmation.

We also mitigated influence of imbalanced data through customizing the weights of loss function, and analyzed via confusion matrix. The final result shows that reweighting gave rise to improvement of recalls of four in five minor classes, which confirms that model implementing reasonable weighted loss function can increase ability to detect minor classes in the data productively. The recall of isolated beta-bridge residue (B) is the only minor class that did not improve. We discussed three possible reasons containing its occupation rate in dataset, the original forecast distribution and the propensity of re-classification. These factors are greatly affected by the original data so different datasets should lead to diverse results.

REFERENCES

1. Heizmann, C. W., Fritz, G., & Schafer, B. W. (2002). S100 proteins: structure, functions and pathology. *Front Biosci*, *7*(1), 1356-68.
2. Fincher, G. B., Stone, B. A., & Clarke, A. E. (1983). Arabinogalactan-proteins: structure, biosynthesis, and function. *Annual Review of Plant Physiology*, *34*(1), 47-70.
3. Lagassé, H. D., Alexaki, A., Simhadri, V. L., Katagiri, N. H., Jankowski, W., Sauna, Z. E., & Kimchi-Sarfaty, C. (2017). Recent advances in (therapeutic protein) drug development. *F1000Research*, *6*.
4. Dekker, F. J., Koch, M. A., & Waldmann, H. (2005). Protein structure similarity clustering (PSSC) and natural product structure as inspiration sources for drug development and chemical genomics. *Current opinion in chemical biology*, *9*(3), 232-239.
5. Waterhous, D. V., & Johnson Jr, W. C. (1994). Importance of environment in determining secondary structure in proteins. *Biochemistry*, *33*(8), 2121-2128.
6. Borguesan, B., Inostroza-Ponta, M., & Dorn, M. (2017). Nias-server: Neighbors influence of amino acids and secondary structures in proteins. *Journal of Computational Biology*, *24*(3), 255-265.

7. Harris, R. K. (1986). Nuclear magnetic resonance spectroscopy.

8. Wuthrich, K. (1989). Protein structure determination in solution by nuclear magnetic resonance spectroscopy. *Science*, *243*(4887), 45-50.

9. Von Dreele, R. B., Stephens, P. W., Smith, G. D., & Blessing, R. H. (2000). The first protein crystal structure determined from high-resolution X-ray powder diffraction data: a variant of T3R3 human insulin–zinc complex produced by grinding. *Acta Crystallographica Section D: Biological Crystallography*, *56*(12), 1549-1553.

10. Muggleton, S., King, R. D., & Stenberg, M. J. (1992). Protein secondary structure prediction using logic-based machine learning. *Protein Engineering, Design and Selection*, *5*(7), 647-657.

11. King, R. D., & Sternberg, M. J. (1990). Machine learning approach for the prediction of protein secondary structure. *Journal of molecular biology*, *216*(2), 441-457.

12. Louis-Jeune, C., Andrade-Navarro, M. A., & Perez-Iratxeta, C. (2012). Prediction of protein secondary structure from circular dichroism using theoretically derived spectra. *Proteins: Structure, Function, and Bioinformatics*, *80*(2), 374-381.

13. Greenfield, N. J. (2006). Using circular dichroism spectra to estimate protein secondary structure. *Nature protocols*, *1*(6), 2876.

14. Wang, G., & Dunbrack Jr, R. L. (2003). PISCES: a protein sequence culling

server. *Bioinformatics*, *19*(12), 1589-1591.

15. Kabsch, W., & Sander, C. (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, *22*(12), 2577-2637.
16. Sander, C., & Schneider, R. (1991). Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins: Structure, Function, and Bioinformatics*, *9*(1), 56-68.
17. Edgar, R. C., & Batzoglou, S. (2006). Multiple sequence alignment. *Current opinion in structural biology*, *16*(3), 368-373.
18. Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic acids research*, *25*(17), 3389-3402.
19. Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *Journal of molecular biology*, *292*(2), 195-202.
20. Jing, X., Dong, Q., Hong, D., & Lu, R. (2019). Amino acid encoding methods for protein sequences: A comprehensive review and assessment. *IEEE/ACM transactions on computational biology and bioinformatics*, *17*(6), 1918-1931.
21. Sønderby, S. K., & Winther, O. (2014). Protein secondary structure prediction with

long short term memory networks. *arXiv preprint arXiv:1412.7828*.

22. Abyzov, A., & Ilyin, V. A. (2007). A comprehensive analysis of non-sequential alignments between all protein structures. *BMC structural biology*, *7*(1), 1-20.
23. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.
24. Hanson, J., Yang, Y., Paliwal, K., & Zhou, Y. (2017). Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics*, *33*(5), 685-692.
25. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.
26. Karystinos, G. N., & Pados, D. A. (2000). On overfitting, generalization, and randomly expanded training sets. *IEEE Transactions on Neural Networks*, *11*(5), 1050-1057.
27. Li, Y., & Shibuya, T. (2015, November). Malphite: A convolutional neural network and ensemble learning based protein secondary structure predictor. In *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (pp. 1260-1266). IEEE.

28. Liu, Y., & Cheng, J. (2016, December). Protein secondary structure prediction based on wavelets and 2D convolutional neural network. In *Proceedings of the 7th International Conference on Computational Systems-Biology and Bioinformatics* (pp. 53-57).

29. Singh, S. K., Bejagam, K. K., An, Y., & Deshmukh, S. A. (2019). Machine-learning based stacked ensemble model for accurate analysis of molecular dynamics simulations. *The Journal of Physical Chemistry A*, *123*(24), 5190-5198.

30. Cheng, J., Liu, Y., & Ma, Y. (2020). Protein secondary structure prediction based on integration of CNN and LSTM model. *Journal of Visual Communication and Image Representation*, *71*, 102844.

31. Provost, F. (2000, July). Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets* (Vol. 68, No. 2000, pp. 1-3). AAAI Press.

32. Nagy, G., Igaev, M., Jones, N. C., Hoffmann, S. V., & Grubmüller, H. (2019). SESCA: predicting circular dichroism spectra from protein molecular structures. *Journal of chemical theory and computation*, *15*(9), 5087-5102.

33. Nagy, G., & Grubmueller, H. (2020). How Accurate Are Circular Dichroism Based Secondary Structure Estimates?. *bioRxiv*.

34. Wang, Zhiyong, et al. "Protein 8-class secondary structure prediction using conditional neural fields." *2010 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2010.

APPENDIX

```
def LSTMModel(x):

    model = Sequential()
    model.add(LSTM(128, dropout=0.1, input_shape=(x.shape[1], x.shape[2]),
                   return_sequences=True, activation='tanh'))
    model.add(LSTM(64, return_sequences=True, activation='tanh'))
    model.add(LSTM(32, return_sequences=True, activation='tanh'))
    #model.add(Dropout(0.2))
    model.add(TimeDistributed(Dense(32)))
    model.add(GaussianNoise(0.1))
    model.add(Activation('relu'))
    model.add(Flatten())
    model.add(Dense(16,activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(8, activation='softmax'))
    opt = keras.optimizers.RMSprop(learning_rate=0.001, decay=1e-6)
    weights = np.array([0.6, 1., 0.8, 1.1, 1.2, 1.1, 1.2, 0.8])
    custom_loss = weighted_categorical_crossentropy(weights)
    model.compile(loss=custom_loss, optimizer=opt, metrics=['accuracy'])

    model.summary()
    return model
```

```
def CNNmodel(x):
    x = x.reshape(x.shape[0], x.shape[1], x.shape[2], 1)
    model=Sequential()

    model.add(Conv2D(32, kernel_size=(3,3), padding='same',
                     input_shape=(x.shape[1],x.shape[2],1),activation='relu'))
    model.add(Conv2D(32, kernel_size=(3,3), padding='same', activation='relu'))
    model.add(MaxPooling2D(pool_size=(1,2)))
    model.add(Dropout(0.2))
    model.add(Conv2D(64,kernel_size=(3,3), padding='same', activation='relu'))
    model.add(Conv2D(64,kernel_size=(3,3), padding='same', activation='relu'))
    model.add(MaxPooling2D(pool_size=(1,2)))
    model.add(Flatten())
    model.add(Dense(256,activation='relu'))
    model.add(Dense(32,activation='relu'))
    model.add(Dense(8, activation='softmax'))
    #weights = np.array([0.6, 1.08, 0.68, 1.017, 1.2, 1., 1.003, 0.73])
    weights = np.array([0.6, 1., 0.8, 1.1, 1.2, 1.1, 1.2, 0.8])
    custom_loss = weighted_categorical_crossentropy(weights)
    model.compile(loss=custom_loss, optimizer="adam", metrics=['accuracy'])

    model.summary()

    return model
```

```python
def CNN_LSTM_model(x):
    model=Sequential()
    #LSTM
    model.add(LSTM(64, dropout=0.1, input_shape=(x.shape[1], x.shape[2]),
                   return_sequences=True, activation='tanh'))
    model.add(LSTM(64, return_sequences=True, activation='tanh'))
    model.add(TimeDistributed(Dense(32)))
    model.add(tf.keras.layers.Reshape((5,32,1)))
    model.add(GaussianNoise(0.1))
    #CNN
    model.add(Conv2D(64,kernel_size=(3,3), padding='same', activation='relu'))
    model.add(Conv2D(64,kernel_size=(3,3), padding='same', activation='relu'))
    model.add(MaxPooling2D(pool_size=(1,2)))
    model.add(Dropout(0.2))
    #flatten and dense
    model.add(Flatten())
    model.add(Dense(256,activation='relu'))
    model.add(Dense(32,activation='relu'))
    model.add(Dense(8, activation='softmax'))
    weights = np.array([0.6, 1., 0.8, 1.1, 1.2, 1.1, 1.2, 0.8])
    custom_loss = weighted_categorical_crossentropy(weights)
    model.compile(loss=custom_loss, optimizer="adam", metrics=['accuracy'])

    model.summary()

    return model
```

```python
def define_stacked_model(members):
    for i in range(len(members)):
        model = members[i]
        for layer in model.layers:
            layer.trainable = False
            layer._name = 'ensemble_' + str(i+1) + '_' + layer.name
    ensemble_visible = [model.input for model in members]
    ensemble_outputs = [model.output for model in members]
    merge = concatenate(ensemble_outputs)
    hidden = Dense(16, activation='relu')(merge)
    output = Dense(8, activation='softmax')(hidden)
    model = Model(inputs=ensemble_visible, outputs=output)
    plot_model(model, show_shapes=True, to_file='model_graph.png')
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model
```