Generating PSSM for Protein Secondary Structure Prediction

William Hogan, Luke Rohrer, Vincent Nguyen
Dept. of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
lrohrer@ucsd.com, vvn012@ucsd.edu, whogan@ucsd.edu

1 Introduction

Proteins are extremely important macromolecules that are used by our bodies in many ways. They provide a wide range of essential functions. For example, proteins can serve as catalysts, speeding up biochemical reactions within our body. The production of adenosine triphosphate (ATP), an organic compound that provides energy to living cells, is made possible by proteins. Proteins also help protect our bodies from pathogens, and antibodies and antigens, which are essential to an immune system, consist of proteins. There are many other functions proteins provide, all of which make them an essential building block of life.

What determines how a protein functions is its structure [11]. By studying protein structure, we can start to understand how a protein works. Knowledge of the function and structure of one protein can often be generalized to other structurally similar proteins [13]. Single mutations within a protein can cause complex diseases. Understanding the position of the mutation, the structure of the mutated protein, and how the mutation affects protein function can lead to a deeper understanding of diseases on a molecular level.

With current technology, analyzing the structure of a protein is a non-trivial problem. Protein structure determination has not enjoyed the same drop in costs compared to protein sequence determination. As of 2018, the cost to determine a protein sequence was $1,000, while the cost to determine a protein structure was $100,000 [4]. This disparity in cost has led to a massive imbalance in available data. There are roughly 140 million sequences stored in the UniProtKB database compared to 150,000 protein structures stored in the Protein Data Bank [9]. The task of protein structure prediction arises from this context—researchers often have access to protein sequence information but lack access to protein structure information. So, researchers have turned to deep learning algorithms for protein structure prediction.

There are two versions of secondary protein structure prediction. Q3, or the 3-state prediction problem, involves characterizing secondary protein structures as belonging to one of three categories: helix (H), strand (E), and coil (C). In 1983, Kabsch and Sander [7] developed finer-grained categories with *Define Secondary Structure of Proteins* (DSSP), or *Q8*, a dictionary consisting of eight distinct protein structures. The Q8 categories are: 3_{10} helix (G), α-helix (H), π-helix (I), β-strand (E), bridge (B), turn (T), bend (S), and others (C) [12]. Q3 is an easier task compared to Q8 since there are only three structures to predict. Q3 prediction is widely considered to be "solved" since Q3 models have achieved performance very close to the theoretical max of 88% accuracy for Q3 classification. There is a 12% variation in secondary structure assignment which leads to a this theoretical maximum. Therefor, we focus our efforts on the more challenging problem of Q8 prediction.

[12] is the current state-of-the-art (SOTA) in Q8 protein structure prediction and their algorithm, and as with other leading algorithms, relies heavily on position-specific scoring matrices (PSSM) to make their structural predictions. Proteins are, at their core, sequences of amino acids. A position-specific scoring matrix is a matrix that corresponds to the frequencies of amino acids at every position of the

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

sequence. Traditionally, these frequencies have been calculated by counting position-specific amino acids from a database of millions of proteins (e.g. PSI-BLAST [1]). Higher scores in the PSSM correlate to residues that appear more at a certain position. This counting approach is extremely time-intensive. The PSSM itself is highly beneficial to the classification task, so samples missing their PSSM become useless when trying to achieve the SOTA. Based on personal experience and discussions online, the generation of PSSM for large amounts of data using PSI-BLAST is infeasible without industrial grade computers. For our final project, we will attempt to develop a generative model to produce PSSMs for protein sequences which, in turn, will be used to augment the training data in the current SOTA models for protein structure prediction. If successful, our approach will expand the amount of data available to train structure-predicting models by millions of samples. To the best of our knowledge, this will be the first attempt to create PSSMs using deep generative techniques.

Using a generative model to create PSSMs is an important and novel idea within the fields of both deep generative models and bioinformatics. Our proposed idea, if successful, would serve as a proprietary example of how deep generative models can be used as an essential processing step within the task of protein secondary structure prediction. As we mentioned above, prior methods that attempt to bridge the gap between the availability of protein sequences and the usability and practicality of such data fall short when it comes to computation and financial cost. Our suggested generative approach would directly address and attempt to solve this major hurdle that is severely limiting the potential progress within this task. We hope to create a model that will simultaneously circumvent the issues regarding PSSM creation, provide evidence that generative models can be successfully used to facilitate progress on downstream tasks, and make available very large and currently unusable datasets to current SOTA methods.

1.1 Contributions

Our contributions in this work are the following:

- We demonstrate that deep generative models can indeed model PSSM data
- We show that the transformer model is superior in generating PSSM data compared to LSTM seq2seq and CNN architectures
- Our results show that, although we did not "solve" this task, the results are promising and, with some refinement, a deep generative model could help make PSSM data more readily available for researchers working on Q8 protein secondary structure prediction

2 Datasets

We use three of the publicly available datasets used in *Prediction of 8-state Protein Secondary Structures by a Novel Deep Learning Architecture*, CB-513 [2], TR-6614 [12], and TR-5534 [6]. CB-513 consists of 513 protein sequences and their corresponding secondary structures for evaluation. For our training and validation split, we use TR-5534 [6] which consists of 5926 sequences with similar format to CB-513. Using TR-5534 for training and validation splits, and CB-513 as the test set mirrors the experiments conducted in [12]. We follow their experimental setup to ensure fair comparison of our results. We generate PSSMs for TR-6614 which is a dataset of protein sequences that does not have PSSM data readily available. Note that the name TR-5334 is given by [12], but is sourced from [6]. We do not know the reason why there are 5926 sequences instead of 5534. All data can be downloaded from [14].

3 Methodology

Our proposed idea is to use deep generative techniques to create PSSM matrices, which will then be used to facilitate the task of protein secondary structure prediction. To do so, we experiment with three deep generative models described in more detail below: a convolutional neural-net (CNN) [INSERT CITATION], a Seq2seq LSTM [8], and a transformer [10].

3.1 Training Procedure and Evaluation

The generative models described below share the same inputs and outputs. Protein sequences are padded to a maximum length of 700. Models use a dense embedding to encode the amino acids of a protein sequence. With the embedded amino acids, the models are trained to generate PSSMs to match the original PSSMs as produced by PSI-BLAST. Outputted PSSMs have a shape (sequence length $\times$ 21), 21 being the number of possible amino acids at any given position. All our models use the TR-5534 dataset randomly split into training and validation sets (80%/20% splits, respectively). To evaluate the generated PSSMs, we compare the results of the secondary structure prediction tasks of the original PSSMs with the generated ones. We further assess our model by generating PSSMs for an additional dataset, TR-6614, and use the augmented data in addition to TR-5534 to train a classification model and compare our result to the baseline classification model from [12].

3.2 CNN

Convolutional models have been shown to be successful in tasks involving sequence to sequence modeling [5]. The first model we tried was a CNN-based sequence to sequence model originally designed for translating text between two languages. This model, as you can see in figure 1 below, consists of an encoder that will encode the the input protein sequence into context vectors, and a decoder that will then decode the context vectors and generate the corresponding PSSM.

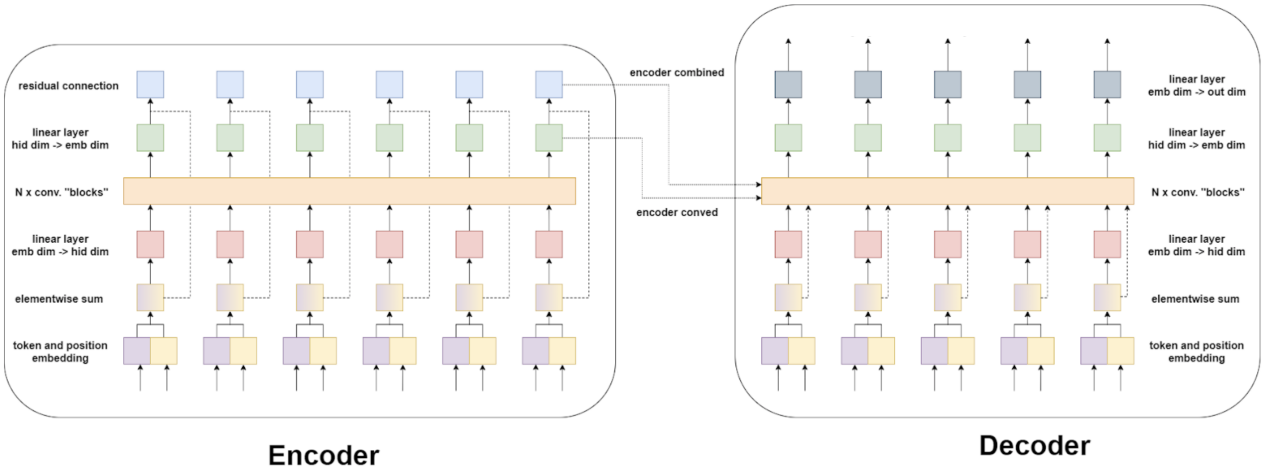

Figure 1: Our baseline CNN sequence to sequence model

In figure 2 below we can see what a single convolutional block in the encoder looks like. The input sequence is first padded so that the output will retain the same number of dimensions as the input. Then, 1024 1D convolutional filters are used to scan across the input and produce our output vectors which are then fed into a set of gated linear activation units. From there the vectors are combined with the embedding of the input token and passed on to the next convoluational block in the sequence. In the final implementation for this model, we used filter sizes ranging from 3 to 25 for each of the convolutional filters (shown as the blue boxes).

3.2.1 Encoder

For each input sequence, the model takes in both the token and position and embeds them separately via two distinct embedding layers. These embeddings are then fed through the encoder to create two context vectors for each token - a *conved* and a *combined* vector. The *conved* vector (shown as the green boxes) is the result of each input being passed through the linear layer and the convolutional blocks, and the *combined* vector (shown as the blue boxes) is the result of summing together the conved vector with the input embedding. These two context vectors are then both fed into the decoder.

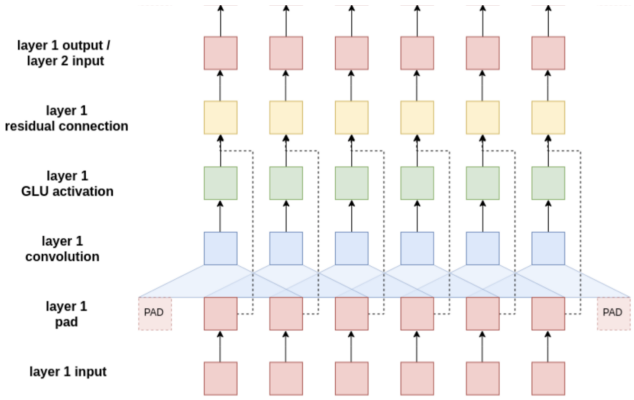

Figure 2: A single convolutional "block" within the CNN sequence to sequence model.

3.2.2 Decoder

The decoder uses the initial token and positional embeddings as well as the *conved* and a *combined* vectors from the encoder to generate the corresponding PSSM output. A main difference here between recurrent decoder structures is that this decoder predicts all output tokens within the target sequence in parallel. As a result, there is no sequential processing or decoding loop.

3.3 Seq2Seq

First introduced in [8], this model uses an LSTM encoder and an LSTM decoder to generate a translation of the original sentence into another language. We adapt the model to generate a PSSM given a protein sequence. We use a baseline unidirectional encoder model, a bidirectional encoder model, and then add an additive attention model on top of the bidirectional model.

3.3.1 Unidirectional Encoder

This model is depicted in Figure 3. For our encoder, we use a single layer LSTM with 250 hidden units. For the decoder, we also use a single layer LSTM with 250 hidden units. We pass the final hidden vector and the final context vector into the decoder to assist generation. During training, the PSSM sequence is fed to the decoder, known as teacher forcing, but during generation the generated PSSM is used. The final PSSM is obtained by concatenating all output hidden states from each timestep from the decoder LSTM. The model is trained using MSE between the generated PSSM and ground truth PSSM. We use a batch size of 60 and the Adam optimizer with a learning rate of 4e-4 and stop training after validation loss rises consistently.

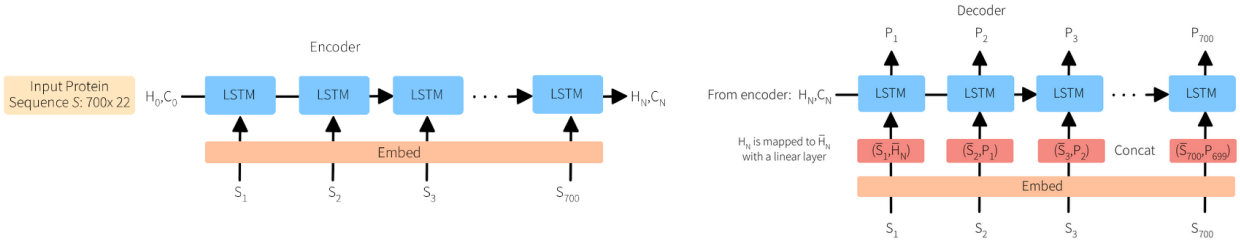

Figure 3: Our baseline seq2seq model

3.3.2 Bidirectional Encoder

This model is depicted in Figure 4 and is very similar to our unidirectional model, but we now use a forward LSTM and a backward LSTM to encode past and future information. Intuitively, protein information should depend on all proteins around it, not just proteins from the past, so we believe this stronger model is more adequate for generation. The two single LSTM both have 250 hidden

units, but the decoder now has 500 hidden units to easily use the concatenation of hidden states from the two encoder LSTMs. Similar to the unidirectional model, teacher forcing is used during training, and the same batch size and optimizer settings are used.

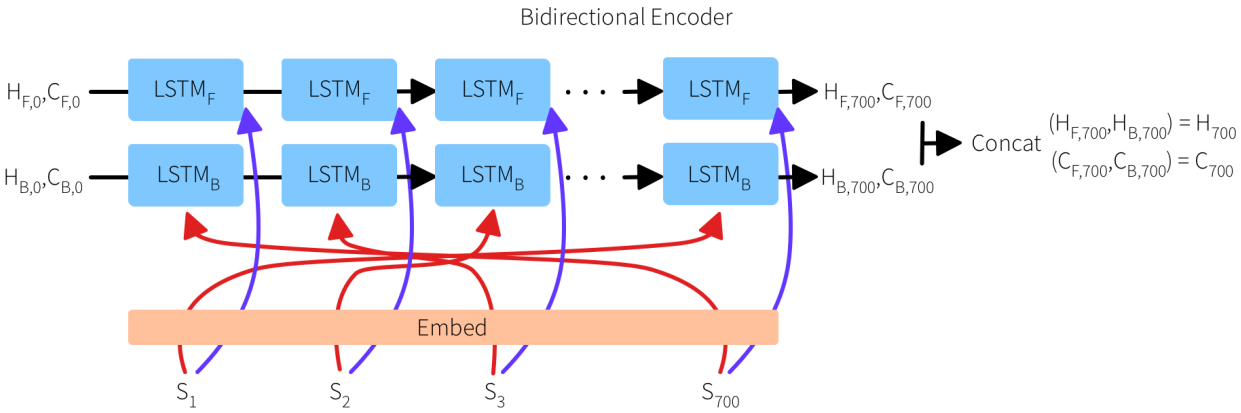

Figure 4: The bidirectional encoder

3.3.3 Additive Attention

As a final seq2seq model, we incorporate additive attention, first introduced in [3], into our encoder and decoder. Intuitively, there may be some parts of the protein sequence that are more important than others. Due to this, we expect the attention model to outperform the previous models. Due to the size of the attention weights, we reduce batch size to 10, encoder hidden unit size to 100, and decoder hidden unit size to 200. The same training routine and optimizer settings are used.

3.4 Transformer

The transformer has been shown to be a very effective model in many sequence-to-sequence applications. It solely relies on attention mechanisms to relate different positions of a sequence in order to compute a representation of the sequence. We modeled our transformer after the one presented in *Attention Is All You Need* [10]. We used 6 stacks of encoders and decoders, each with 8 attention heads, and an encoding dimension of 512. PSSMs were normalized to help reduce the time needed for training. For training we used a batch size of 16, a learning rate of 1e-5, and a dropout of 0.1. We implemented early stopping based on the validation loss. We used an Adam optimizer and mean-squared error for the loss function which allowed the model learn to generate the continuous values of the PSSM. For the complete model and hyperparameter settings, please reference the link to the repository provided in Section 4.3.

4 Experimental Results

4.1 Baseline

The following are baseline results for Q8 protein secondary structure prediction using the datasets described in Section 2 and no augmented data. The results were obtained by building the convolutional, residual, and recurrent neural network (CRRNN) model proposed in [12]. Training on TR-5534 and testing on CB-513, we obtain 70.21% $\pm$ 0.20 accuracy which matches the performance reported in [12].

Table 2 is a confusion matrix showing how well the baseline model predicts each class in the Q8 task. Referencing the counts of Q8 classes provided in Table 3, we can see the baseline model does well predicting common classes such as *H* and *L* but it struggles to predict rare classes such as *B* and *S*. These results highlight the need for more usable training data (which we hope to produce with our work) and higher performing Q8 models.

Table 1: Baseline experimental results on Q8 protein secondary structure prediction using no augmented data.

Model	Training Data	Test Data	Q8 Accuracy %
CRRNN	TR-5534	CB-513	70.21 $\pm$ 0.20

Table 2: Confusion matrix from the baseline Q8 predictive model. Predictions are on the y-axis and ground-truth labels are on the x-axis.

	B	E	G	H	I	L	S	T
B	**28**	263	11	81	0	662	41	95
E	5	**15055**	54	251	0	2184	129	338
G	1	148	**973**	763	0	635	25	587
H	1	231	277	**24172**	0	821	41	614
I	0	2	0	24	**0**	1	0	3
L	9	2817	183	707	0	**12374**	506	1324
S	0	876	147	550	0	3603	**1414**	1726
T	0	470	382	1621	0	1742	205	**5593**

4.2 Generation Results

4.2.1 Sanity-check

To confirm our models were reasonably generating PSSMs, we conducted a sanity-check experiment. For this, we compared three versions of training data which were used to train the Q8 prediction model. The first version featured a TR-5543 training set with original PSSMs (generated from PSI-BLAST). The second version featured a TR-5543 training set with PSSMs generated from our transformer model. The third version featured a TR-5543 training set with PSSMs removed. These experiments are visually depicted in Figure 5.

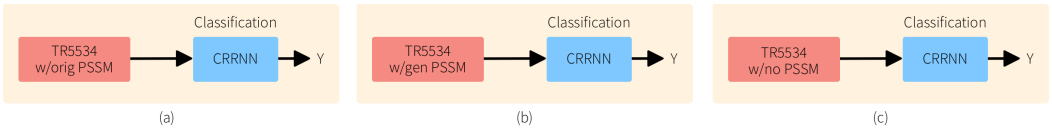

Figure 5: Sanity-check experiments we conduct to demonstrate our models are generating reasonable PSSMs. (a): Training a Q8 classifier on TR-5534 with original PSSMs. (b): Training a Q8 classifier on TR-5534 with generated PSSMs. (c): Training a Q8 classifier on TR-5534 with no PSSMs.

The results in Table 4 show that our model was indeed successful at generating PSSMs that aided in the Q8 classification task. The training set with generated PSSMs performed similarly to the training set with the original PSSMs. Lastly, the poor performance from the experiment with no PSSMs highlights the importance of PSSMs in Q8 prediction and, thus, the importance of this work. PSSMs are indispensable to Q8 prediction and making them more readily available should help facilitate future Q8 research efforts.

4.2.2 Augmented Training Results

Beyond assuring our models were capable of generating informative PSSMs, we also wanted to use generate PSSMs to augment training data to improve the state-of-the-art baseline in Q8 secondary structure prediction. To do this, we constructed an augmented data training pipeline which is depicted in Figure 6. In the first stage, we trained our models to generate PSSMs using the TR-5534 training set. The results in Section 4.2.1 show that these efforts were successful. With a newly trained generative model, we moved to the second stage—generating PSSMs for TR-6614, a dataset which is lacking PSSM data. We then moved to the third stage where we combined the TR-6614 dataset containing generated PSSMs with the TR-5534 dataset containing original PSSMs to train the Q8 classifier.

Table 3: Counts for each Q8 class in the CB-513 test set.

Label	B	E	G	H	I	L	S	T	Total
Count	1180	17994	3132	26143	30	17904	8310	10008	84701
%	0.014	0.212	0.037	0.309	0.0004	0.211	0.098	0.118	1.00

Table 4: Sanity-check experiments using varied training sets for Q8 protein secondary structure prediction.

Training Variation	Q8 Accuracy %
TR-5534 with original PSSMs (baseline)	70.21 ± 0.20
TR-5534 with generated PSSMs	70.03 ± 0.13
TR-5534 with no PSSM	37.54

Table 5 contains the results from our augmented training experiments. The original baseline model for Q8 classification was established using only the TR-5534 dataset containing original PSSMs. We hoped that our experiment with the augmented data which combined TR-5534 (original PSSMs) and TR-6614 (generated PSSMs) would achieve higher performance given the increased size of the training data. Unfortunately, none of the experiments using PSSMs generated from our generative models were able to surpass the baseline. Our transformer came close and its results are promising but it will require further refinements to surpass the baseline.

We did have success outperforming the baseline model trained using the TR-6614 dataset without PSSMs and TR-5534 with PSSMs. Three of our models, namely the LSTM bidirectional, LSTM bidirectional + attention, and the transformer, were able to generate PSSM data for TR-6614 which led to better performance for Q8 classification. Although this was not our primary goal, this experiment does show that the PSSM data generated by our models is capable of augmenting training data and improving Q8 prediction without PSS0M.

Within the experiments with the LSTM seq2seq model, we see each increase in model complexity—unidirectional to bidirectional, and no attention to added attention—came with a boost in performance. This matches our expectations since bidirectional LSTMs with attention are known to perform well in sequence-to-sequence tasks [8]. The CNN, however, underperformed relative to our other models. We believe this is due to the lack of attention mechanism in the CNN. The CNN cannot easily attend to non-adjacent amino acids across the inputted protein sequences. It, instead, must learn long-range dependencies through stacked convolutional layers. Anoher downfall of the CNN model was that it produced all output token predictions (which is a vector of dimension 700×21) in parallel. Although CNNs have seen some success in other sequence-to-sequence tasks, we believe the generation of a sequence of PSSMs with 21 features per time-step and long-range dependencies is too challenging for the CNN architecture.

4.3 GitHub Repositories

For our project, we have two repositories: one for the PSSM generative task and one for the protein secondary structure prediction task. We decided to separate each task00 to increase the modularity of the project. The PSSM generative task can be seen as a distinct pre-processing data augmentation step to the downstream task of protein structure prediction. Each model can be found at the following links:

PSSM Generator (CNN sequence to sequence):
`https://github.com/wphogan/q8-protein-structure-prediction/tree/cnns2s`

PSSM Generator (Transformer):
`https://github.com/wphogan/transformer_pssm_generator`

PSSM Generator (seq2seq):
`https://github.com/wphogan/q8-protein-structure-prediction/tree/s2s`

Q8 Protein Structure Prediction:

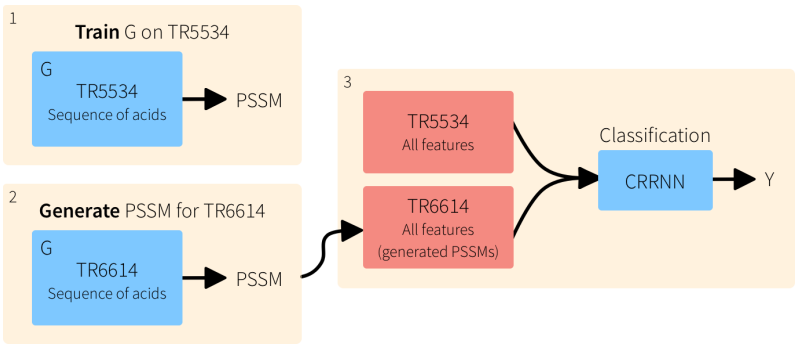

Figure 6: Augmented training pipeline. Step 1: Use TR-5534 to train a generator to generate PSSMs. Step 2: Generate PSSMs for TR-6614 (a dataset which is missing PSSMs). Step 3: Use TR-6614 with generated PSSMs to augment the training of a Q8 classifier.

Table 5: Results from augmented training experiments. Models with an asterisk feature an augmented Q8 training with protein sequences from TR-6614 containing generated PSSMs combined with the TR-5534 dataset containing original PSSMs

Model Type	**Q8 Accuracy %**
Baseline CRRNN (no PSSM)	70.21 ± 0.20
Baseline CRRNN (TR-5534 w/original PSSM)	$\mathbf{70.21 \pm 0.20}$
Baseline CRRNN (TR-5534 w/original PSSM + TR-6614 w/no PSSM)	69.1238 ± 0.1356
CNN-based sequence to sequence*	39.1 ± 0.10
LSTM seq2seq unidirectional*	69.0745 ± 0.15
LSTM seq2seq bidirectional*	69.2113 ± 0.43
LSTM seq2seq bidirectional + attention*	69.24 ± 0.31
Transformer*	$\mathbf{70.1464 \pm 0.37}$

`https://github.com/wphogan/q8-protein-structure-prediction`

5 Conclusion and Discussion

In this project, we were successful in generating PSSMs that were capable of informing Q8 protein secondary structure prediction. However, we were unable to improve the state-of-the-art baseline for Q8 prediction using our generated PSSMs to augment training. Our experiments with different models highlights the efficacy of the transformer for PSSM generation. Future works may benefit from focusing on optimizing the transformer for this task. We are hopeful that more work in this area will lead to a model that makes PSSMs readily available to researchers in this area of work. If performance can improve, a pre-trained transformer could potentially replace the costly PSI-BLAST algorithm.

References

[1] Stephen Altschul, T.L. Madden, Alejandro Schaffer, Jianmin Zhang, Zhengh Zhang, W.E. Miller, and D.J. Lipman. Gapped blast and psi-blast: a new generation of protein databases search programs. *Nucleic acids research*, 25:3389–402, 10 1997.

[2] Zikrija Avdagic, Elvir Purisevic, Samir Omanovic, and Zlatan Coralic. Artificial intelligence in prediction of secondary protein structure using cb513 database. *Summit on translational bioinformatics*, 2009:1–5, Mar 2009. 21347158[pmid].

[3] Yoshua Bengio Dzmitry Bahdanau, Kyunghyun Cho. Neural machine translation by jointly learning to align and translate. 2014.

[4] Yang et al. Sixty-five years of the long march in protein secondary structure prediction: the final stretch? *Brief Bioinformatics*, 19:482–494, 05 2018.

[5] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning, 2017.

[6] Olga G. Troyanskaya Jian Zhou. Deep supervised and convolutional generative stochastic network for protein secondary structure protein. *arXiv*, Mar 2014.

[7] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.

[8] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[9] Irena Roterman-Konieczna Tomasz Smolarczyk and Katarzyna Stapor. Protein Secondary Structure Prediction: A Review of Progress and Directions. *Current Bioinformatics*, 15:90–107, 01 2020.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[11] Wikipedia. Protein structure — wikipedia, 2020. [Online; accessed 26-April-2020].

[12] Buzhong Zhang, Jinyan Li, and Qiang Lü. Prediction of 8-state protein secondary structures by a novel deep learning architecture. *BMC Bioinformatics*, 19(1):293, Aug 2018.

[13] Freddolino Zhang. COFACTOR: improved protein function prediction by combining structure, sequence and protein–protein interaction information. *Nucleic Acids Research*, 45(W1):W291–W299, 05 2017.

[14] Jian Zhou. Cb513 dataset.