Sequential Protein Secondary Structure Prediction using Recurrent Neural Networks

Hassan Nishat **Owais Zahid** **Yufei Ren**

CSC413
University of Toronto
{hassan.nishat, owais.zahid, philip.ren}@mail.utoronto.ca

Abstract

Link to project Colab Notebook

This project is dedicated to the task of predicting the secondary structure of proteins from their primary sequences, which plays an important role in drug discovery and structural biology. Given that the determination of a protein's 3D structure through X-ray crystallography or Nuclear Magnetic Resonance (NMR) is a costly and time-intensive process, our goal is to develop a deep learning model capable of accurately inferring secondary structures in a more efficient manner, finally enhancing the computational determination of protein structures. Our proposed approach utilizes a Recurrent Neural Network (RNN) with bi-directional Long Short-Term Memory (LSTM) units, specifically designed to handle sequential data and capture the long-range dependencies between amino acids which are vital for accurate structure prediction. The result of our study shows that around 55% accuracy in both training and validating is achieved using bi-directional LSTM model with the model structure under Model Architecture section of our report.

1 Introduction

The interest in predicting protein secondary structure from amino acid sequences arises from the desire to streamline and accelerate this discipline of structural biology. By predicting secondary structures accurately, researchers can gain insights into protein function and stability without relying on expensive and labor-intensive experiments. The importance of this task is underscored by its potential to revolutionize how we approach protein engineering, drug discovery, and our broader understanding of cellular mechanisms.

This project proposes a computational approach to predict protein secondary structures from primary sequences. To achieve this, we choose to use Recurrent Neural Network (RNN) architecture utilizing Long Short-Term Memory (LSTM) units (Tsukiyama et al., 2021)[5]. LSTMs are specifically engineered to capture long-range dependencies by mitigating the vanishing gradient problem commonly associated with standard RNNs. By doing this, LSTM makes them ideally suited for modeling the sequential nature of protein strands where distant amino acids can influence folding patterns (Tsukiyama et al., 2021)[5]. Our model architecture is designed to include multiple layers of LSTMs to enhance learning capacity, each followed by batch normalization to accelerate training and combat overfitting. Additionally, we incorporate bidirectional configurations and dropout layers strategically placed between LSTM layers to induce regularization and improve the model's understanding of the context in sequence data.

The model will be trained to predict secondary structure of proteins, with simplification of the conventional eight-state (Q8) classification for improved manageability and computational efficiency.

By focusing on the primary sequence data, the model aims to bypass the need for expensive structural determination methods, while seeking to improve upon the current Q8 prediction accuracy of approximately 70%.

2 Background and Related Work

The primary structure of a protein is defined as the sequence of amino acids linked together to form a polypeptide chain. (Sanvictores; Farci., Biochemistry, Primary Protein Structure) [1]. A peptide is a short chain of amino acids (typically 2 to 50) linked by chemical bonds (called peptide bonds). A longer chain of linked amino acids (51 or more) is a polypeptide. The proteins manufactured inside cells are made from one or more polypeptides. (Hull, National Human Genome Research Institute) [2]

A protein's secondary structure is just its localized shape, and there are eight categories or labels, of secondary structure that we will be classifying: loops and irregular elements, β-strand, α-helix, β-bridge, 3-helix, π-helix, turn and bend respectively. More details on how to classify the aforementioned structure is included in the **Data** section.

We will be using a neural network architecture with an LSTM RNN layer (Long Short Term Memory) for this task, and note that there have been attempts in the past to use RNN's for protein structure prediction, for example in the following notebook: Protein Structure from Secondary Seq - 2022 [3] however there are a couple key differences. The model in the notebook uses the RMSProp algorithm for optimization and the Softmax activation function for the final dense layer, whereas we use the Adam optimizer and the ReLU activation function respectively. More details on our model's architecture can be found under the aforementioned section of our proposal.

3 Data

The dataset we have chosen for our project is entitled "Protein Secondary Structure - 2022" [4] and was published by the user KIRKDCO on Kaggle. This dataset is an updated version of "Protein Secondary Structure", another dataset posted to Kaggle by the user Alfrandom. The updated version of the data contains information taken from the RCSB Protein Data Bank as of August 6th, 2022 and includes 164,619 unique protein strands. Protein structures in this dataset are represented by the letters; C, E, H, B, G, I, T, and S. These letters represent loops and irregular elements, β-strand, α-helix, β-bridge, 3-helix, π-helix, turn, and bend respectively. The dataset includes seven columns of information for each entry. The columns are made up as such:

- **pdb_id**: an id that can be used to locate the entry on the RCSB PDB website
- **chain_code**: a code used to keep track of proteins which consist of multiple peptides
- **seq**: the primary sequence of the peptide
- **Sst8**: the 8-state secondary structure
- **Sst3**: the 3-state secondary structure
- **Len**: the length of the peptide
- **Has_nonstd _aa**: a boolean showing if the peptide has any nonstandard amino acids

The key columns we will be looking at during our study are seq and sst8. Seq represents our input variable and sst8 will be the value that we aim to predict. Seq describes the given peptide sequence and sst8 is a form of protein secondary structure calculated from the peptide's 3D coordinates which are derived from X-ray images. Sst8 classifications of secondary structures of peptides consist of eight categories of structures [Table 1].

Table 1: Protein Categories and Structure

SST-8	Structure
E	β-strand
B	β-bridge
H	α-helix
G	3-helix
I	π-helix
C	loops and irregular elements
T	Turn
S	Bend

4 Model Architecture

We started by encoding data with One-Hot Encoding for Primary Sequence and Secondary Sequence (SST8): Each amino acid in the protein sequence is represented as a one-hot encoded vector. In one-hot encoding, each amino acid is represented by a vector where the length of the vector equals the number of possible amino acids (the vocabulary size). Each position in the vector corresponds to a specific amino acid. The vector contains all zeros except for a single one at the position representing the specific amino acid in question.

4.1 Model Layers

The model layers consist of the following:

1. Embedding Layer
2. 2 Bidirectional LSTM Layers
3. First Fully Connected Layer
4. Second Fully Connected Layer

After padding and encoding our input sequence, we pass it into the embedding layer, followed by the LSTM layer and into the first fully connected layer. We also use the ReLU activation function in between the first and second fully connected layers.
Upon leaving the final fully connected layer, we have a one-hot encoding for each character in each sequence in every batch, corresponding to the model's predicted SST8 value at that position in the sequence. Finally, we evaluate the accuracy by comparing the ground truth label's value with the model's value at each position in every sequence for each batch, and predict the total percentage of the sequence that the model predicted accurately as our metric for a successful prediction.

4.2 Training and Validation Settings

For our model parameters, we have set the vocab size, embedding dimension, hidden and output dimensions, and number of layers to 21, 60, 70 and 8, 2 respectively. Our training choice has been limited due to constraints imposed by computational resources at our disposal, which will be discussed further in the **Limitations** section, however we set 8 as the output dimension as we have a total of 8 classes to predict for each character in the output sequence.

Finally, our output for an input of our chosen batch size of 10 and **unpadded** length L inputs per batch would be $10 \times L \times 8$. An illuminating representation of the following can be found in the following figure.

5 Model Figure

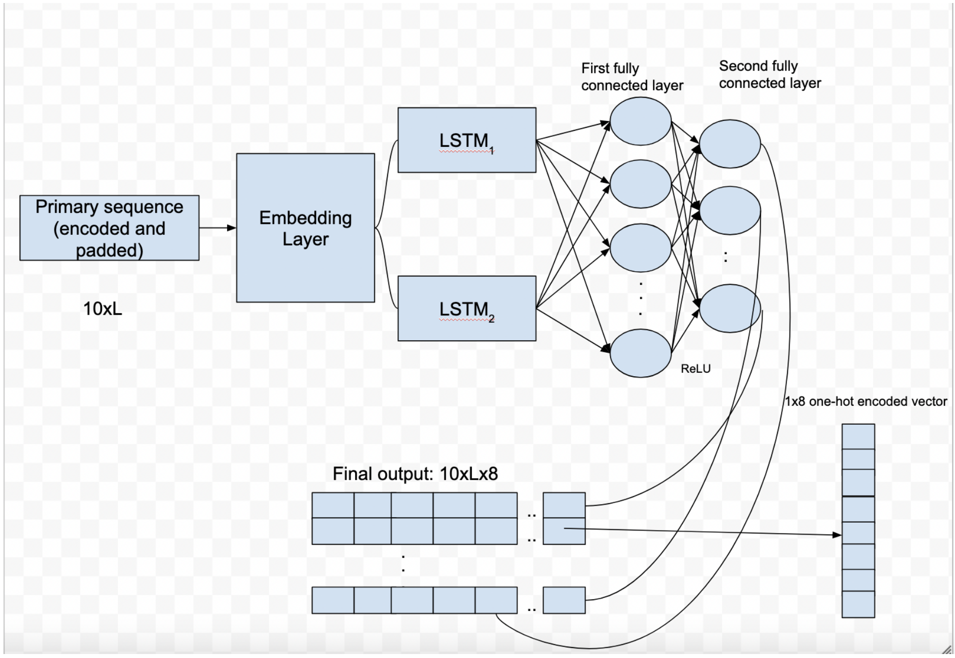

Figure 1: LSTM Model Figure

6 Results

After determining the optimal settings for our model an accuracy of about 55% was achieved. These graphs describing the accuracy and loss over training were obtained after fully training our model.

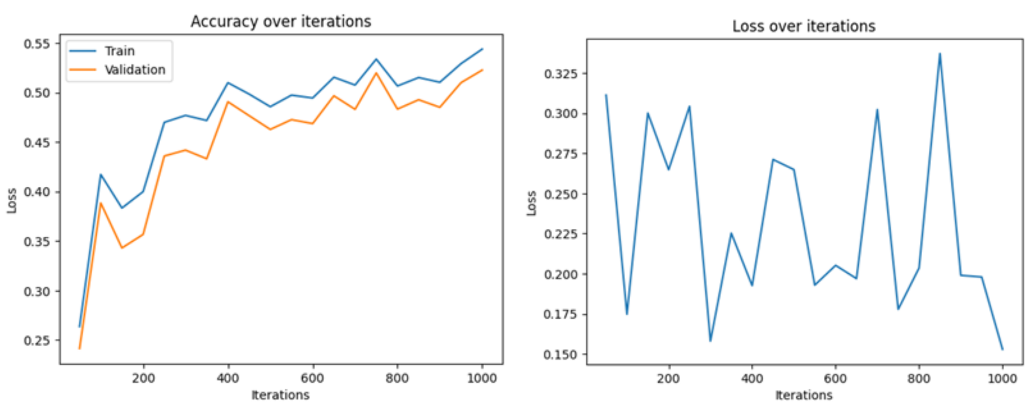

Figure 2: Accuracy and Loss graphs

To show what the output of the model looks like below are the predictions compared with the actual sequences of the first five protein structures in our test set.

```
Input Sequence: GKSPTEVLLELIAEASGTTREEVKEKFLKELRKGKSPTEVLLELIAEASGTTKEEVKEKFLKELSFGKSPTEVLLELIAEASGTTKEEVKKKFWKELSL
Predicted Sequence: TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTHHHHHHHHH
Actual Sequence: HHSHHHHHHHHHHHHHTSHHHHHHHHHHHHHHTTHHHHHHHHHHHHHHHTHHHHHHHHHHHHHHHTTHHHHHHHHHHHHHHHTHHHHHHHHHHHHHHTH
```

As a baseline model, we chose the model provided along with the Kaggle dataset. The creator of the dataset chose to use a prebuild tensorflow model named “Sequential”. The model is set up with

an embedding layer that has an input dimension with the size of the number of unique sequences in the dataset, an output dimension of size 128, and a max length of each structure set to 300. The model also has a Bidirectional GRU layer which allows the model to process the input sequence in previous and future time steps providing the model with more context. Finally the model also has a TimeDistributed layer which applies a Dense layer to each time step of the model.

To train the baseline model, firstly each protein sequence was encoded by dividing each sequence into k-mers. K-mers are subsequences of the total protein sequence of a certain length k, which is the length of the sequence itself. Each protein is divided into all of the possible k-mers from beginning to end. For example, a sequence of EBH would be split into: E, EB, and EBH. These new sequences are then tokenized before training.

The baseline model calculates accuracy for training in a similar way to how we do. The accuracy percentage is determined by comparing the similarity each individual predicted protein structure to the expected value rather than accounting for how many predictions were identical to their expected value. This is the same approach taken in our model, however one difference is that the baseline model takes into account both sst-3 and sst-8 secondary structures.

Here is a snippet of the code used to create this model.

```
model = Sequential([
    Embedding(input_dim = n_words, output_dim = 128, input_length = maxlen),
    Bidirectional(GRU(units = 64, return_sequences = True, recurrent_dropout = 0.1)),
    TimeDistributed(Dense(n_ssts, activation = 'softmax'))])
```

Our baseline model showed an accuracy ranging between 65% and 85% depending on the size of training data. This is a higher accuracy displayed by our model, the reasons for this are further discussed in our discussion and limitations.

7 Discussion

In our discussion of the model's performance, several key points emerge, particularly when comparing its training and validation accuracy with its performance on test data. The model achieves around 55% accuracy on both training and validation sets, which suggests a moderate level of learning. However, our model performs poorly on test data, the output is heavily biased towards predicting a specific class (in this case, 'T') for most of the sequence, with occasional occurrences of another class (like 'H'). This pattern of prediction, where the model overwhelmingly favors one or a few classes, is indicative of a few potential issues:

First of all, model underfitting, the model might not be learning the complex patterns in the data effectively. This could be due to insufficient model complexity, inadequate training (e.g., not enough epochs, inappropriate learning rate), or issues with the data representation. It may also be attributed to the fact that we have not used the **Softmax** activation function on the model output after passing through the final fully connected layer (after all, we are doing multiclass classification for all characters within each sequence), however this did not seem to make much improvement in the overall performance of our model, given our limited choice of parameters due to limitations discussed further in the next section.

The second issue we need to think about is generalization ability, the discrepancy between the model's performance on training/validation data and test data suggests a generalization issue. While the model has learned patterns present in the training set, it struggles to apply these learnings to unseen data. This could be due to underfitting, or due to differences in the distribution of the training and test datasets.

Thirdly, the moderate accuracy on the training and validation sets suggests that the model has some predictive power, but it might not be complex enough to capture the full intricacies of protein sequences. Alternatively, the method of data representation (e.g., encoding of amino acids) might not be adequately capturing the necessary biological information.

To solve the above issues and to improve our model performance, one possible direction in our future study would be exploring more complex models or architectures, such as transformer-based models,

might yield better results. Additionally, integrating external biological knowledge, such as protein-protein interactions or evolutionary information, could provide a more holistic understanding. In conclusion, While the model demonstrates some ability to learn from the training data, its performance on unseen test data highlights the challenges in protein sequence prediction. These challenges include the need for better generalization, more effective data representation, and possibly more complex modeling techniques. Future work should focus on addressing these challenges, potentially through more advanced encoding methods, enhanced model architectures, and ensuring the quality and representativeness of the training data.

8 Limitations

There were several limitations pertaining to the model parameters and computational resources available. Note that we had initially planned to use a dropout layer as we had anticipated our model to overfit on the dataset in our project proposal, however this turned out to not be the case as our architecture led to underfitting in most cases. Additionally, we did not have the most optimal selection of model parameters (i.e embedding dimension, number of layers and hidden dimension of the LSTM layer, etc) and this is largely due to computational constraints imposed on us by Google Colab. This also hindered us from extending our trained model to attempt transfer learning on an adjacent task.

Each free Google Colab session allows for only a single NVIDIA T4 GPU with 15GB VRAM. Thus, we had to be conservative when setting the parameters for our model (embedding size, number of hidden layers, etc) as ambitious values for the aforementioned params led to us running out of memory. This would not be the case for smaller input sizes, however each sequence in any batch was atleast 400 characters long, and so the large input size combined with computational restraints were bottlenecks in our search for optimal model parameter values. Also, the free edition of Google Colab Cloud Instances that we used have a limited number of runs per day per user, considerably limiting our scope for grid search.

Another possible limitation is our encoding method for the sequences themselves, We used one-hot encoding to turn each amino acid inside the sequence into it's integer representation, but this method doesn't really capture all the complex details about how proteins work. For example, it doesn't show how amino acids interact with each other in 3D space or how they change after the protein is made (which can really affect what the protein does). Also, our method doesn't consider how proteins are related to each other through evolution or how they vary. This is important because these relationships can tell us a lot about what a protein does and why it's important. To make our approach better, we could try using more advanced ways of representing proteins that take into account things like the chemical properties of amino acids or how they are arranged in 3D space. We could also try to include information about how proteins have changed over time and how they interact with other molecules in the body. But these improvements would make the model more complex and harder to work with, so we'd need to find a good balance between making the model better and keeping it simple enough to use.

9 Ethical Considerations

One ethical consideration to be made about our study is the integrity of the data. We use data collected from the RCSB Protein Data Bank, which is an open access digital resource. Since it is an open access platform, its data is consisted of uploaded data from researches. This could call into question the integrity of the data as it is possible that someone could upload tampered or harmful data which could lead to a corrupted model making incorrect predictions. RCSB-PDB, however, is a large and trusted source and claims to thoroughly examine all uploaded data. This issue is still worth to keep in mind throughout our study.

Another consideration could be the misuse of our created technology. Our goal in creating this technology is to create a way to determine secondary structures of proteins without taking X-ray images, allowing processes such as drug discovery to become quicker and less resource consuming. Such a technology, however, could also be used for the opposite reasons. Our technology could

potentially be used to design harmful proteins, so it is important to ensure that our model will be used in line with our intentions.

Finally, another ethical consideration that we could implement in the future is that of having a transparent model. Transparency is a key feature in ensuring that a neural network is ethical as it allows for the predictions made by the model to be explained. This is especially important in the case of predicting protein structures. For example, if some medication were to be developed using protein structures predicted by a machine learning model and someone using the medication was harmed in some way, it would be important to understand why exactly the model made the prediction it did so that we could understand what went wrong and how the issue can be solved.

10 Conclusion

In conclusion, we found that our implementation of an LSTM model to predict secondary Sst-8 protein structures from the given primary structure shows some promise of having predictive power, however does not exceed or improve upon models that are already out there. This was seen as our model showed a predictive accuracy of about 55% whereas our baseline model shows accuracy within a range of 65% and 85%. These findings could be for a number of reasons that we have discussed. Namely, we believe that our model was not complex enough to tackle the task at hand.

In future works we believe that the concerns raised by this project can be addressed. A more complex model would be chosen, and we would avoid google colab to ensure we do not run into the same issues again. Transparency and explainability would also be features that could be added when furthering this work.

Although our model did not perform as well as we had initaly anticipated, we believe that there is still work to be done in protein sequence prediction. Our work has shown that using LSTM models are not as efficient as other models being used and that the main focus going forward should be on trying other model architectures to try and improve what results are being seen.

11 References

1. Sanvictores; Farci., Biochemistry, Primary Protein Structure: https://www.ncbi.nlm.nih.gov/books/NBK564343/
2. Hull, National Human Genome Research Institute, Peptides: https://www.genome.gov/genetics-glossary/Peptide
3. Protein Structure from Secondary Seq - 2022: https://www.kaggle.com/code/kirkdco/protein-secondary-struct-from-seq-2022
4. Protein Secondary Structure - 2022: https://www.kaggle.com/datasets/kirkdco/protein-secondary-structure-2022
5. Tsukiyama, S., Hasan, M. M., Fujii, S., & Kurata, H. (2021). LSTM-PHV: Prediction of human-virus protein–protein interactions by LSTM with word2vec. Briefings in Bioinformatics, 22(6). https://doi.org/10.1093/bib/bbab228