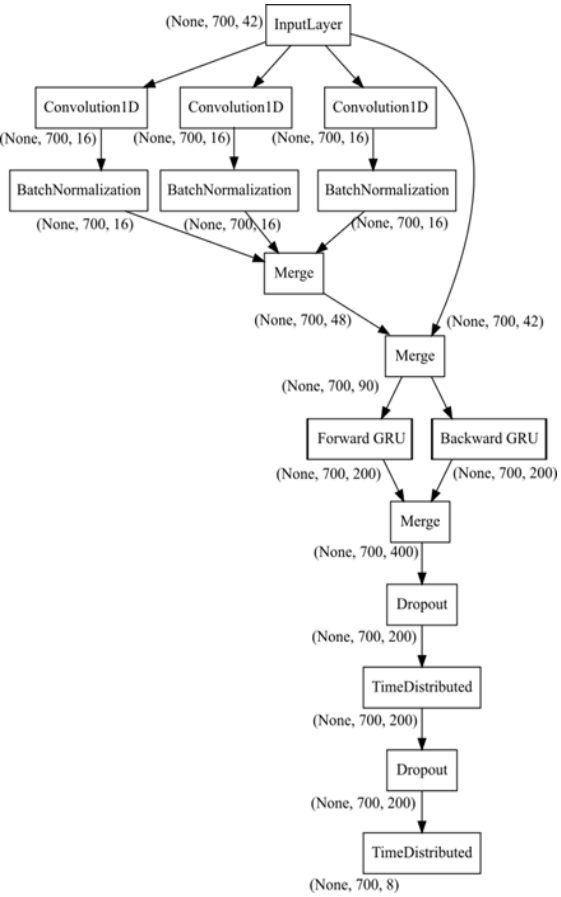
Protein Secondary Structure Prediction using Deep Learning with Convolutions and Bidirectional Gated Recurrent Unit

M. Rizky Luthfianto, Afiahayati
Department of Computer Science and Electronics, Universitas Gadjah Mada
m.rizky.l@mail.ugm.ac.id, af1a@ugm.ac.id

Protein secondary structure prediction is an important problem in bioinformatics because the three-dimensional structure of a protein determines the function of the protein itself. Meanwhile, protein is the main functional units in living organisms and are involved in many biological processes in cells.

According to Graves (2008), sequence labeling is all tasks where a sequences data is transcribed with discrete labels. Furthermore, protein secondary structure prediction can be viewed as a sequence labeling problem, because each of amino acids is labeled with its own secondary structure.

In this research, we implemented a Deep Learning model for protein secondary structure prediction which combines Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).



In this model, there are three convolutional layers with different filter lengths which are arranged in parallel to extract local features. This method is better than splitting the sequence with a sliding window because these layers can learn different features and does not complicate the preprocessing.

Because long-range dependencies are proven to exist in amino acid sequences as discovered by Zhou & Troyanskaya (2014), we used Bidirectional Gated Recurrent Units which is an RNN to capture the global context features.

We use three datasets which are CullPDB, CullPDB-Filtered, and CB513. CullPDB-Filtered is a dataset which has no more than 25% similarity with proteins in CB513. Each sample in datasets has 700 timesteps and 42 features.

For the convenience of processing and implementation, the length of all protein amino acid sequences in these datasets are normalized to 700, which means sequences originally shorter than 700 are padded with zeros. To keep the computations correct, we do masking which is multiplying the padding of timesteps with zero, so the padding will not ruin the loss computation and the backpropagation process.

Before we evaluate our model on the test set, we searched the optimal hyperparameters in the validation set. The searched hyperparameters are the optimal epoch, learning rate, Batch Normalization usage, gradient clipping, and L2 regularization.

Each experiment is implemented on Theano and done on a Nvidia GTX 980Ti with a batch size of 128 samples which runs about 100 seconds per epoch, which took 4 hours and 10 minutes for 150 epochs.

After evaluating on the validation set, we conclude the optimal hyperparameters for our model are 83 epochs, learning rate 0.0001, with Batch Normalization enabled, without Gradient Clipping, and without L2 regularization.

The resulting model is able to achieve both Q8 accuracies of 69.7% on CullPDB test-set and 66.9% on CB513 which higher than 66.4% by Zhou and Troyanskaya (2014) in their previous research. After trying three different initial weights, we also found that Batch Normalization consistently speeds up the convergence up to 3x, because our model can converge in 83 epochs or 86 epochs, rather than 249 epochs. While L2 regularization doesn't always help reducing model overfitting.

In the future, we like to explore the addition of Embedding layers and Conditional Random Fields to increase the accuracy, and switching out Bidirectional GRU for Quasi-RNN for a faster learning process.