Sequential Deep Learning Methods for Protein Function Prediction

Dan Amaranto, Akash Kadel, Brenton Arnaboldi
NYU Center for Data Science, Capstone Project

Objective

Predict the Go Terms (functions) associated with a protein using only amino acid sequence.

Introduction

New protein sequences are identified at a very fast pace, but deeper understanding of the identified sequences takes time to verify. Machine learning can help identify the functions of new protein sequences. The Critical Assessment of Functional Annotations (CAFA) project invites researchers to create models for this task. The most successful CAFA submissions often use a wide variety of features, including protein interactions, mass spectrometry, gene interactions, and so on. By contrast, we aim to tackle the protein prediction problem with a single type of data and relatively few training examples.

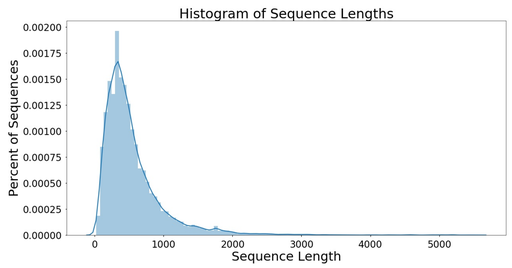
Data

Inputs: Variable length sequences of type string, where each letter represents one of 22 amino acids in a protein sequence (e.g. "MPAMQPUXPLQ"). The average string is about 582 characters for human proteins and 536 for yeast.

	Human	Yeast	Combined
Training Set	9751	3447	13198
Validation Set	3871	963	4834
Test Set	1647	206	1853

Table 1: Number of proteins in each set

Outputs: This is a multilabel classification task. Each sequence can predict any of 153 total GO functional terms.



Models

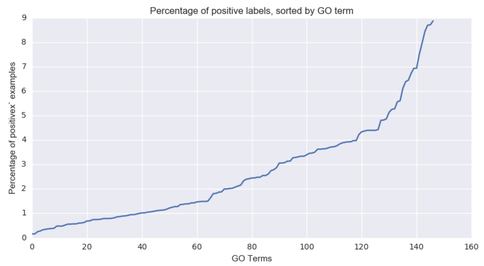
Prior to training, the individual amino acids were randomly initialized into embedding vectors. We compared models with and without k-mers (bigrams and trigrams of amino acids), each of which was also randomly initialized to an embedding vector.

FastText: FastText was developed to train word embeddings for text classification tasks. We adopt it here to train amino acid embeddings for the multilabel classification task.

Convolutional Neural Net (CNN): Our baseline deep learning model was a CNN, which is fast, memory-efficient, and location invariance. A key question is whether neural nets that consider sequence order will perform better than those that do now, hence CNN was necessary as a benchmark.

Long Short Term Memory Unit (LSTM): Recursive Neural Nets read over sequences, and hence they base their predictions off of sequence order. Amino acid sequencing is an essential part of their structure and functions, so we would assume that RNNs will be the preferred method. First we look at LSTM RNNs, which excel at capturing relationships between distant elements in a sequence.

Gated Recurrent Unit (GRU): We trained our model using a variation of a gated RNN, the GRU. GRUs are faster and more memory-efficient than LSTMs. The performance of GRUs relative to LSTMs varies based on task. Unlike LSTM, it exposes its entire hidden state, which can be useful for our data problem (due to data shortage).



Results - Summary

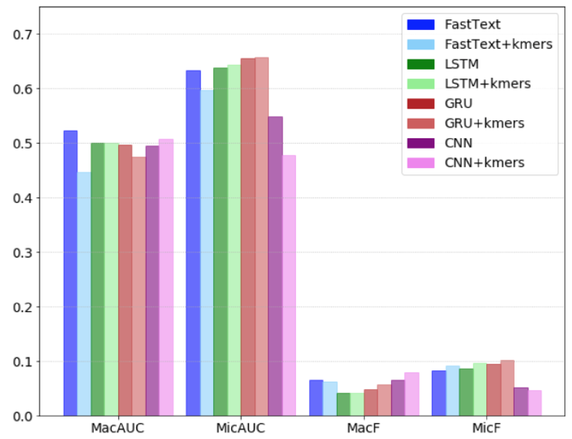


Figure 1: Summary Results - Human

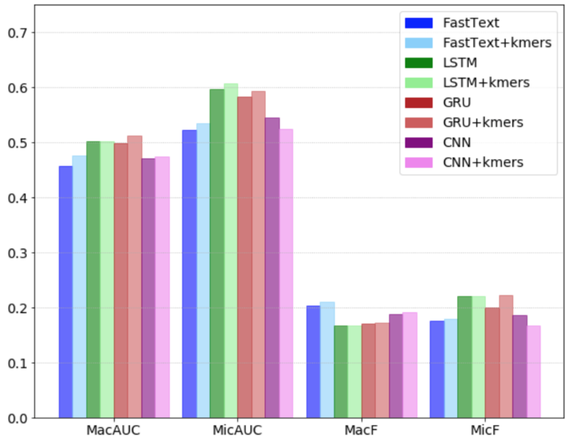


Figure 2: Summary Results - Yeast

Methods and Performance: For each model we tuned embedding dimensions, the size of the hidden layer, learning rate, and L2 penalty. We also regularized with early stopping. Compared to more robust models with large training sets and various inputs, our models did not perform well. This is not all that surprising, given that deep learning methods generally require lots of data. We re-ran our models on combined datasets of both human and yeast sequences, but it did not lead to a performance improvement.

Results by Go Term

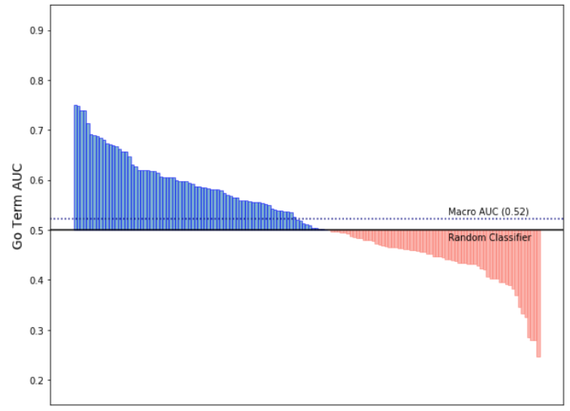


Figure 3: AUCs by Go Term (Human FastText). Note that our model is better at predicting certain Go Terms than others.

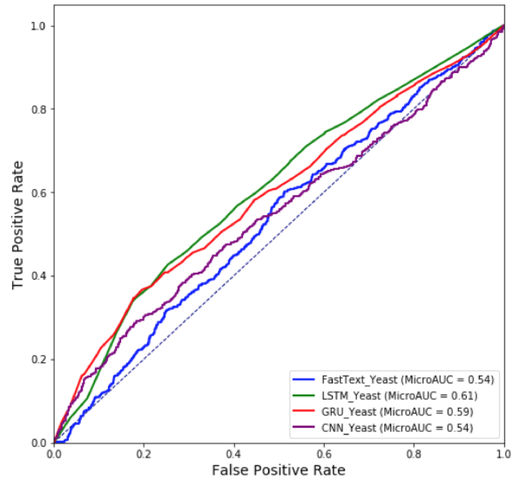


Figure 4: Receiver Operator Curves (ROCs) for Yeast

Conclusion/Future Work

Our biggest problems were lack of training data and highly imbalanced classes. While down-sampling the majority class might mitigate the imbalanced classes problem, it would decrease our already small amount of data. Protein prediction is a difficult task; state-of-the-art models incorporate many data types and train on large sets of data.