BMI203: Final

Laura Shub

March 2020

Q1. *Describe the machine learning approach in detail. This will include, for an ANN, a description of the network structure of your encodings of inputs and output. For an SVM, you will need to discuss input encoding as well as kernel function choices, etc... This will also include a description of the representation you have chosen for the input DNA sequences.*

To classify the sequences, I used a simple ANN with 1 input layer, 1 hidden layer, and 1 output layer. Each base in the input sequences was represented as a bit string, encoding both identity and purine vs pyrimidine. These were 6 bits long, and each sequence was 17 bases long, leading to an input layer size of 102. The hidden layer, and the output layer contained one node, giving a real value between 0 and 1, with 1 designating a transcription factor binding site. Each layer used a logistic activation function, and I used mean squared error as my error function.

Q2. *How was your training regime designed so as to prevent the negative training data from overwhelming the positive training data?*

I prevent negative training data from overwhelming the positive by both oversampling positive examples and undersampling negative examples. At each epoch, I select n samples with replacement from each class. This combined class-balanced dataset is then used as the training set for that epoch to prevent the model from consistently predicting non-binders.

Q3. *What was your stop criterion for convergence in your learned parameters? How did you decide this?*

There are multiple stop conditions. The first is if the number of epochs is reached, and the second is if the loss drops below 0.005. This second criteria was reached by inspection of the loss over time on both the autoencoder problem and running the classifcation problem with various hyperparameters.

Q4. *Describe how you set up your experiment to measure your system's performance.*

In testing the autoencoder, I checked reconstruction on an 8x8 identity matrix.

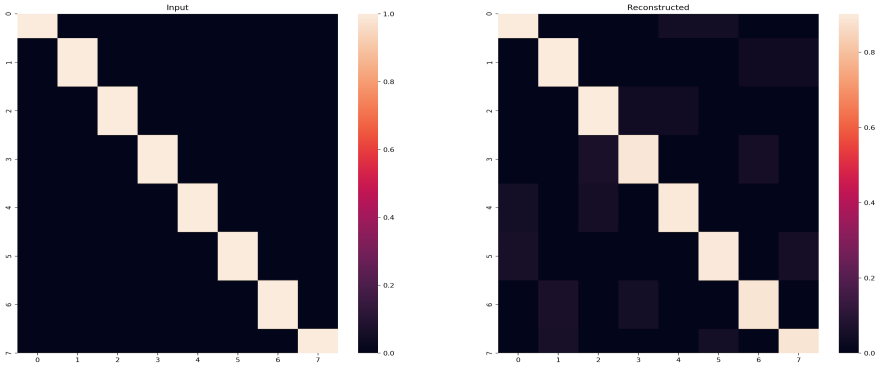

Figure 1: Visualization of input and reconstruction of identity matrix by 8 x 3 x 8 autoencoder

To measure the performance of my classifier, I use 5-fold cross validation by holding out one fifth of the training data, training the model, then evaluating the model on the holdout set. This was performed

for a number of different hyperparameters to determine the optimal set. The final model was trained on the full dataset using the best hyperparameters.

Q5. *What set of learning parameters works the best? Please provide sample output from your system.*

I tested a variety of hyperparameters including learning rate, number of epochs, batch size, number of layers, number of neurons, and sample size. The full list, as well as the average validation loss is in loss_dict.txt.

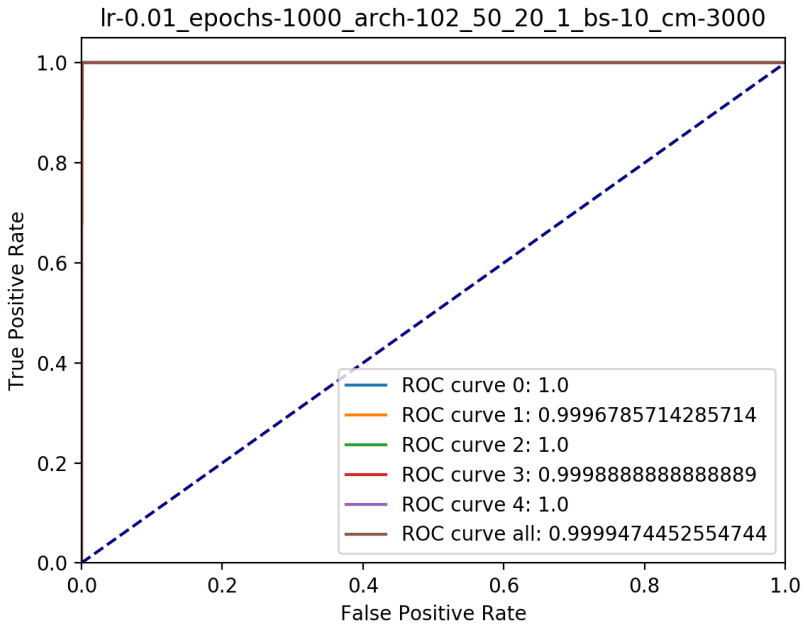

Figure 2: Cross validation and full training set ROC for best hyperparameters

The best hyperparameters that I found were a learning rate of 0.01, 1000 epochs, and an architecture with 102 input nodes, 2 hidden layers with 50 and 20 nodes each. This resulted in an AUC of very close to one for all 5 CV validation sets.

Q6. *What are the effects of altering your system (e.g. number of hidden units or choice of kernel function)? Why do you think you observe these effects?*

One issue that I saw is that the performance increases as the number of hidden layers or the number of nodes in the hidden layers increased. The 102-50-20-1 architecture outperformed both the 102-50-1 and 102-20-1 architectures, especially at lower learning rates. This might suggest that a good local minimum was difficult to find with a smaller number of tunable parameters, and perhaps the number of training epochs that I used were not enough to converge to that point.

Q7. *What other parameters, if any, affect performance?*

Another parameter that I tested was my sample size for each epoch, or the number of positive and negative examples I use in updating the weights. Increasing the number of samples lowered the overall loss as one might expect, but it also significantly increased the training time.

Github: `https://github.com/laurashub/bmi203final`.