Week 06

Work done

The week has been spent working on alternate neural network models in hopes of obtaining better results than the previously used Zhang et al model.

Since the input is already preprepared as a (50,96) matrix, that is 50 words with 96 features each, we have attempted three types of ad-hoc models that take as input a 2 dimensional matrix. The first is a pure multi-layer perceptron matrix.

Layer (type)	Output Shape	Param #
input_25 (InputLayer)	(None, 50, 96)	0
flatten_6 (Flatten)	(None, 4800)	0
dense_41 (Dense)	(None, 2000)	9602000
dense_42 (Dense)	(None, 1000)	2001000
dense_43 (Dense)	(None, 500)	500500
dense_44 (Dense)	(None, 200)	100200
dense_45 (Dense)	(None, 50)	10050
dense_46 (Dense)	(None, 50)	2550
dense_47 (Dense)	(None, 1)	51

Total params: 12,216,351
Trainable params: 12,216,351
Non-trainable params: 0

The second is a hybrid of dense and 2 dimensional convolutional matrix.

Layer (type)	Output Shape	Param #
input_26 (InputLayer)	(None, 50, 96)	0
reshape_14 (Reshape)	(None, 50, 96, 1)	0
conv2d_23 (Conv2D)	(None, 16, 32, 100)	1000
conv2d_24 (Conv2D)	(None, 4, 8, 200)	320200
conv2d_25 (Conv2D)	(None, 1, 1, 400)	2560400
global_max_pooling2d_11 (Glo	(None, 400)	0
dense_48 (Dense)	(None, 400)	160400
dense_49 (Dense)	(None, 200)	80200
dense_50 (Dense)	(None, 100)	20100
dense_51 (Dense)	(None, 50)	5050
dense_52 (Dense)	(None, 25)	1275
dense_53 (Dense)	(None, 10)	260
dense_54 (Dense)	(None, 5)	55
dense_55 (Dense)	(None, 1)	6

Total params: 3,148,946
Trainable params: 3,148,946
Non-trainable params: 0

And the third is an LSTM and Dense model:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 50, 96)            0
_________________________________________________________________
lstm_1 (LSTM)                (None, 300)               476400
_________________________________________________________________
dense_1 (Dense)              (None, 300)               90300
_________________________________________________________________
dense_2 (Dense)              (None, 200)               60200
_________________________________________________________________
dense_3 (Dense)              (None, 100)               20100
_________________________________________________________________
dense_4 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_5 (Dense)              (None, 25)                1275
_________________________________________________________________
dense_6 (Dense)              (None, 10)                260
_________________________________________________________________
dense_7 (Dense)              (None, 5)                 55
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 6
=================================================================
Total params: 653,646
Trainable params: 653,646
Non-trainable params: 0
_________________________________________________________________
```

We run these architectures for 200, 300, and 400 epochs on our 10 000 training examples, then test them on our 2500 examples. All all the models were trained for all epochs, as it would have been redundant. The results are as follows.

Accuracy	Training(200 Epochs)	Test	Training(300 Epochs)	Test	Training(400 Epochs)	Test
Dense	90.522718	69.305187	92.125058	69.437116		
Conv	71.040547	72.647315	71.040547	72.647315	98.006839	63.280565
LSTM	**71.040547**	**72.647315**			**71.040547**	**72.647315**

All three types of networks provide better results than the Zhang et al implementation. We at least have experimental support that other models are more useful, even if this puts us back at the start of the project. Now all that remains is further improvements of these models.

Problems

The main problem immediately apparent from the above results is a convergence to certain values. Whatever model is used, the values converge to around 70%, or they push forward to an overfit.

To do for next week:

Since we cannot augment the data in a deterministic manner, our main option is to reanalyze the dataset. A few options exist and will be attempted:

- use word vectors with larger number features than our current 96
- further reduce noise by eliminating the proteins we do not care about
- adjust the learning rate and implement regularizations