Week 07

Work done

The week has been spent investigating the convergence issues. In essence, different types of networks produce the same accuracy results for a variable number of epochs.

Our initial preprocessing assumed that we would concatenate the proteins we sought(e.g. PROTEIN3 and PROTEIN4) at the front of the abstract. However, we chose to reinterpret the dataset, and replace the target proteins with keywords such as PROTEINA and PROTEINB, and substitute the rest of the proteins with randomly generated word. By this procedure we hoped to obtain a more clear dataset the network might have an easier time processing. Encoding it in a (50,200) matrix, that is (words_in_abstract,wordvector_dimensiont) with scispacy, we ran this reprocessed dataset with a Dense, CNN and LSTM network. Results have not been encouraging.

Accuracy	Train 200 epochs	Test 200 epochs	Train 300 epochs	Test 300 epochs	Train 400 epochs	Test 400 epochs
Dense	0.7108074426651001	0.7324486374855042	0.7108074426651001	0.7324486374855042	0.8947165608406067	0.6476883292198181
CNN	0.7108074426651001	0.7324486374855042	0.7108074426651001	0.7324486374855042	0.7108074426651001	0.7324486374855042
LSTM	0.7108074426651001	0.7324486374855042	0.7108074426651001	0.7324486374855042	0.7108074426651001	0.7324486374855042

We still have convergence issues. A more in-depth analysis suggested that scispacy encoded keywords PROTEINA and PROTEINB both resulted in a zero vector, as do unknown words, and thus the network was unable to generalize. This is something the researcher had not expected. One could make the assumption that embedding layers are functions that compute word vectors based on available data. Instead, they seem to be a direct map from a word to a vector. As such, the embeddings need training on our keyworded dataset as well.

Given that scispacy is largely an external tool that cannot be included in the keras functional API model, we have instead opted for word2vec, glove, and fasttext respectively as trainable embedding layers in the hopes that our network will learn to differentiate and classify based on our keywords. We ran each of these embeddings through our three types of networks (Dense, CNN, LSTM) for 200 epochs each, but to no avail, as the networks still converged to largely the same values.

Accuracy	word2vec train	word2vec test	glove train	glove test	fasttext train	fasttext test
Dense	0.822012722492218	0.613869845867157	0.8314412236213684	0.610873281955719	0.8401000499725342	0.5903253555297852
CNN	0.7107946872711182	0.7324486374855042	0.7107946872711182	0.7324486374855042	0.7107946872711182	0.7324486374855042
LSTM	0.7107946872711182	0.7324486374855042	0.7107946872711182	0.7324486374855042	0.7107946872711182	0.7324486374855042

Problems

The researcher has hit the problem with everything he could, from various techniques to process the dataset, various embeddings, various network architectures, different epochs etc. Not only has none of these methods produced satisfying results, the results converge to ***identical values*** regardless of variance.

To do for next week:

The researcher is discouraged and at a complete loss as to what to do next. Two months of reading and coding efforts have resulted in nothing of worth. He can only hope the project advisor takes some time to look at the latest code updates and offer some ***practical code suggestions***, as yet another indecipherable research article by a chinese researcher will certainly be of no use given how little time is left.