Sprint 01

Work done

Spent most of the sprint studying genetics concepts along with a semi-meta-analysis article on the tools used to apply Natural Language Processing(NLP) to the task of protein to protein to protein interactions(PPI) since these were completely and largely unknown to the researcher.

Scope

It appears that the aim of our current project is text mining applied to corpora of scientific articles studying protein to protein interactions. A number of challenges become readily apparent upon studying the literature.

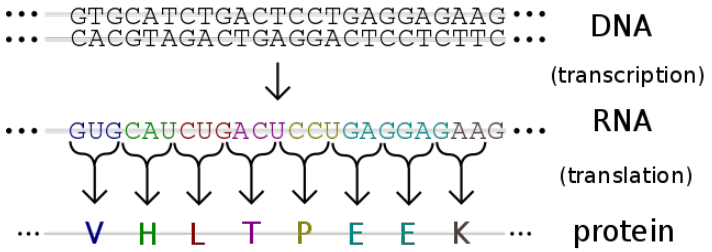
The first is extracting the keywords(proteins) from the text, especially given the large diversity of unstandardized texts. A second challenge is the to study the relations between these keywords in a computationally efficient manner. Finally, we must output predictions regarding these interactions.

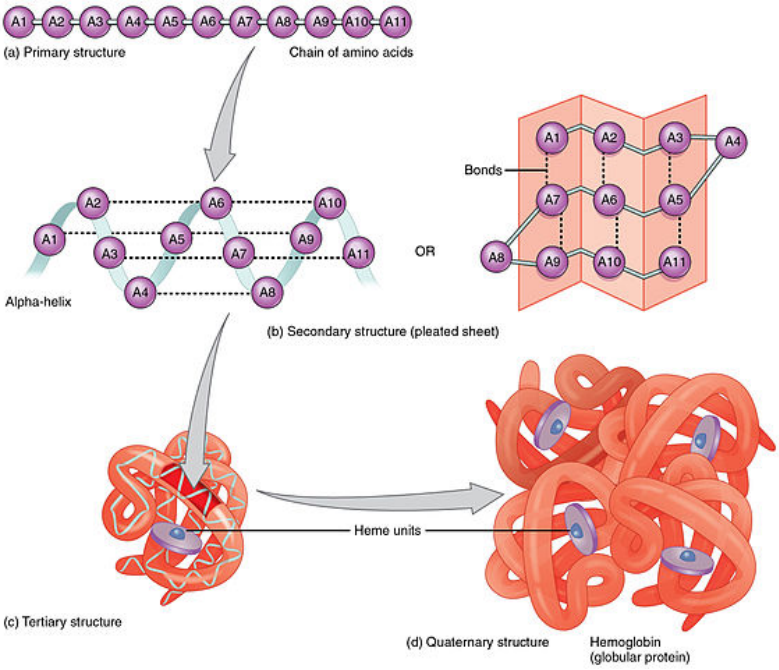
Genetics Background

While proteins themselves will be the central object of study, understanding them requires also understanding other parts of the genetic complex. A short summary of the knowledge collected is as follows.

The collected genetic material of an organism is called a genome or genotype. The way it is expressed in an organism as it develops, especially considering external factors, is the phenotype. At a base level, the DNA is a backed up genetic code base with four key elements(nucleotides) in various arrangements. The double spiral splits into halves and unfolds into RNA, which bonds with the free organic molecules in the cell nucleus. Codons, units of three nucleotides, work to program aminoacid compounds. Multiple codons called create a chain of such amino acids which is known as a protein. In an organism, a protein serves multiple purposes, from chemical signals to support structures.

In short, we can create analogies of DNA as a factory spitting out structures based on programmed code. Although the process is likely to be a lot more imprecise and random due to the inherent organic and evolutionary nature of the elements, hopefully this analogy will be useful going forward.

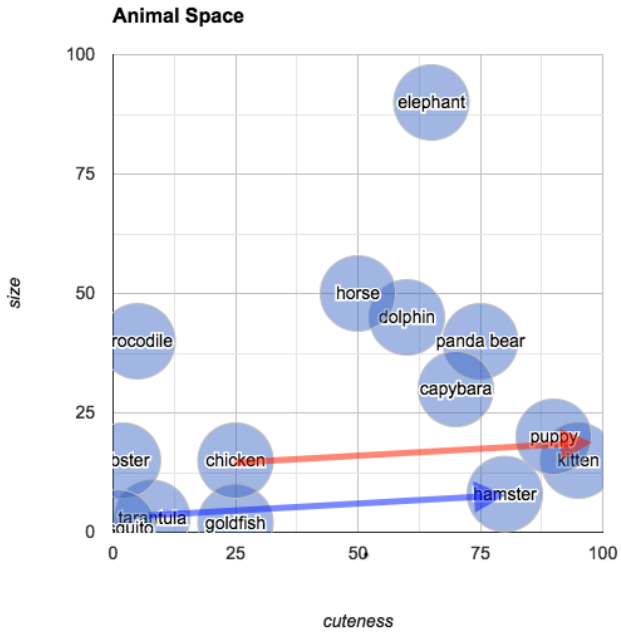




Natural Language Processing

On the other hand, a bit less time has been spent reviewing natural language processing methods, as the researcher is familiar with these methods already.

Word embedding are methods where strings in a text are mapped out to mathematical vectors based on various methods. A common one is the contextual word vector, where each element in the vector measures the frequency of all other common words surrounding the target word. This is known as the bag of words or n-grams(where in is the number of surrounuding word) and relies on methods such Tf-idf to counts the frequency of words in a corpus of text and decide which is the most important one. Other methods include unsupervised learning methodologies. Finally, we have refinement techniques such as stemming(a raw cutting of the word's terminations such as -ind, -ed) and lemmatization(a more fine tuned version of stemming which can instead return the infinitive of a verb).



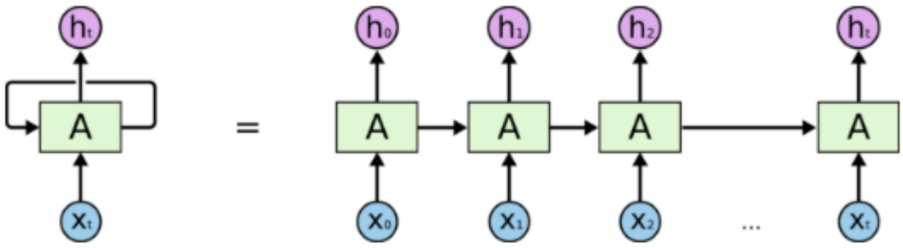
Putting the two areas together, it seems like proteins, being amino acid chains, can be represented as word vectors, allowing us to use natural language processing methods to classify proteins(classifiers), study interactions and bonds(vector similarities), and produce new protein sequences(LSTM).

Zhang et al. in their "Deep Residual Convolutional Neural Network for Protein-Protein Interaction Extraction" provide a rudimentary starting point into the field, albeit a highly technical one. Their article beings with a short presentation of text mining methods used.

The first type of text mining is by kernel-based PPI extractors, in other words *classic parsing*, using parse trees along with combinations of other tools. These methodologies are computationally and algorithmically heavy, as well as case specific. Given the average biochemist is not a Nobel Laureate in literature, we can expect unorthodox sentencing which might cause trouble for a traditional parser, which is likely to be more suited to machine languages.

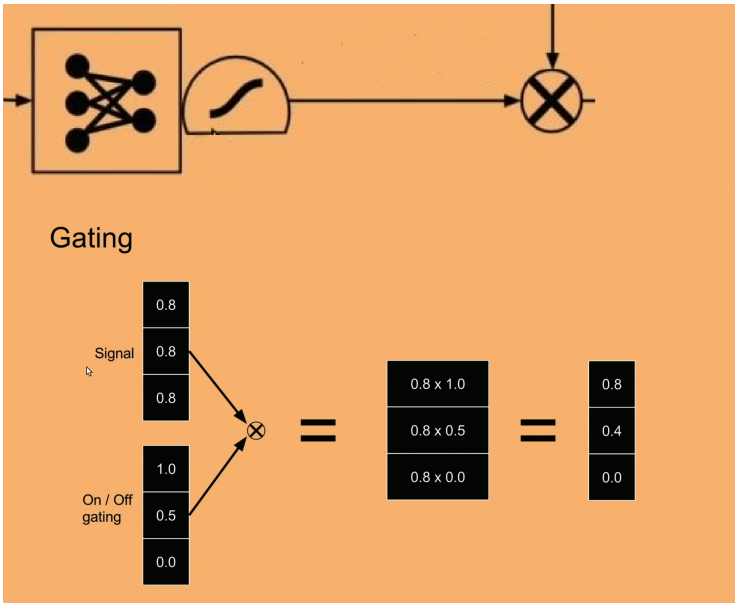
The second category are machine learning methods. While there are more obscure methodologies in practice such as support vector machines, perhaps it is best to focus on more commonly used methods: Recurrent neural networks(specifically LSTMs) and convolutional neural networks.

We'll start with the first. A recurrent neural network has the peculiarity that it takes two inputs: one is the traditional input, while the other is the previous output of the network, fed back into itself.

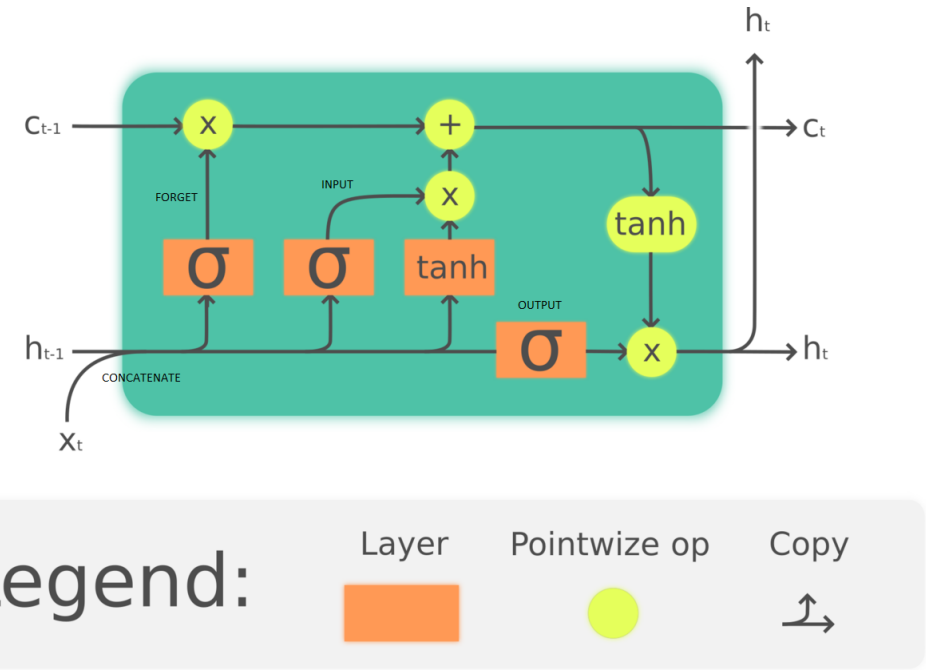


An unrolled recurrent neural network.

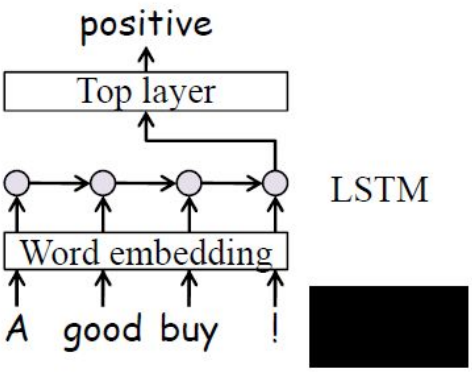
This adds a temporal component to neural networks, allowing for memory over time, providing context to the input. The LSTM is merely a more specialized type of RNN, and in fact consists of multiple networks. Perhaps the most important of its standard component is the gate, with is a neural network combined with an activation function and an element-wise multiplication.



By combing these gates, we have a unit with memory that is not only selective but adaptive as well.

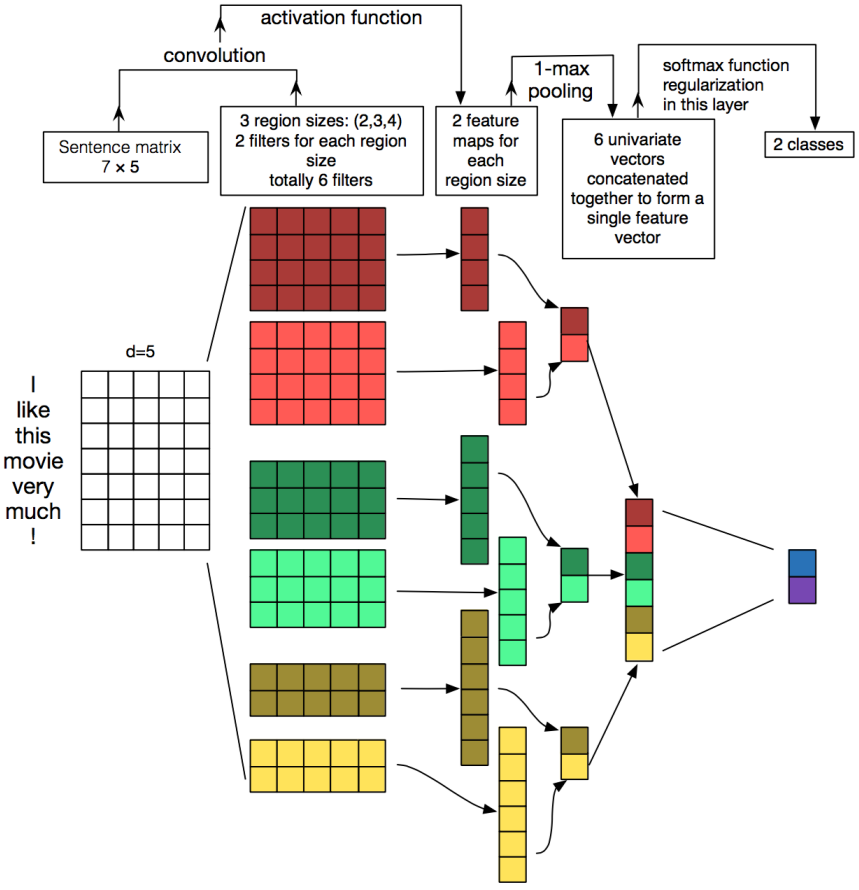


The literature suggests that we feed word vectors into the LSTM one by one, which in the end should output the desired result. Below is a sample architecture with a classifier.

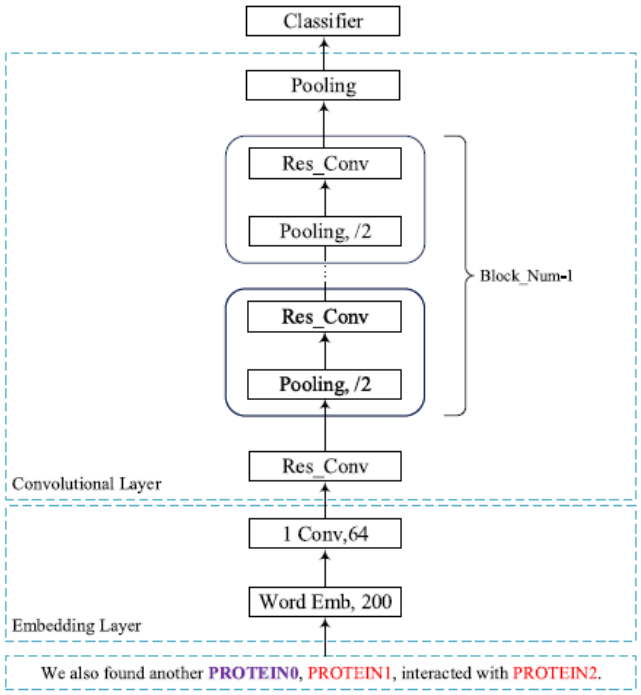


One advantage that LSTMs provide compared to other methods is inter-sentence dependencies, allowing us to train and feed the whole text word by word into the unit and gets results. A major downside of this methodology is the massive computational power required, since we are in essence running multiple neural networks. The parameters alone are daunting.

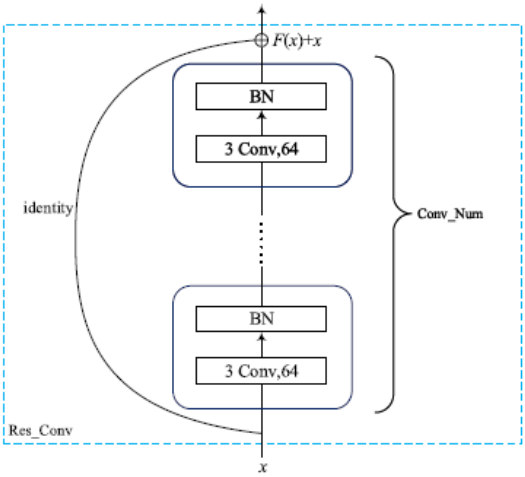
What Zhang et al. propose instead is a CNN based network. The general approach is to embed each word, producing vectors of size d. Thus, a sentence of n words becomes a n x d matrix. 2D Convolutions can naturally be applied to such matrices, producing feature maps which can then be pooled into output vectors. An example architecture with multiple convolutions for high and low feature extraction can be seen below.



Specifically, the researchers first hone in on the protein words, replacing them with keywords that stand out (PROTEIN1, PROTEIN2, PROTEIN3...) in a vocabulary table. Afterwards they pass the sentence matrix through several specialized CNN layers called residual convolutional blocks. The main aspect of these blocks is combining the processed output with the previous input, thus avoiding backpropagation infinitesimal issues such gradient vanishing or explosion. The end result is a softmax classification vector, where each entry is a combination of two proteins in the vocabulary table. The model is summarized below.



(a) Our Model



(b) Residual Convolutional Block

Zhang et al. offer metrics which suggest this model is on at least par with LSTMs and other methods. Considering its much lower computational needs, it is advantageous to choose this model. This might be initially counterintuitive, as we expect that a neural network would have to process a word one by one, as the LSTM architecture would. Perhaps the efficiency of this deep CNN can be explained at an intuitive level by the following sentence:

Yuo cna porbalby raed tihs esaliy desptie teh msispeillgns.

As John Von Neumann pointed out in his unfinished book on the human mind, our thought processes are likely probabilistic, approximating the words we scan over. Indeed, human fast reading techniques are based on this process but at the level of a sentence or even a page. Biological fast readers sacrifice precision for an overall feel of what the text implies, which suffices for simple concepts. The deep CNN model offered by Zhang et al. likely does the same, the parameters estimating the relation between the keywords(proteins) at a higher level.

Overall, this is the model we should follow.

Problems

The article itself is highly technical, offering only some mathematical formulations with no code examples or practical implementations, which are likely left to the reader. A few outstanding issues are:

1. How the word tabulation is implemented, especially with tools such as nltk, spacy, or the keras embedding layer.
2. Sentences are likely to vary in size. How then do we insure the CNN kernels apply universally to all sentences.
3. A scientific text consists of multiple sentences. What do we do about about inter sentence dependency? How do we decide which sentences to study?
4. The softmax classifier output is dynamic. It is a 2-combination of k proteins. How do we insure connectivity and universality with the rest of the model?

Work to do

The next phase is likely to be a study of the implementation of the vocabulary table and interactions with it. Regardless of the actual methodology chosen, it will probably use it.