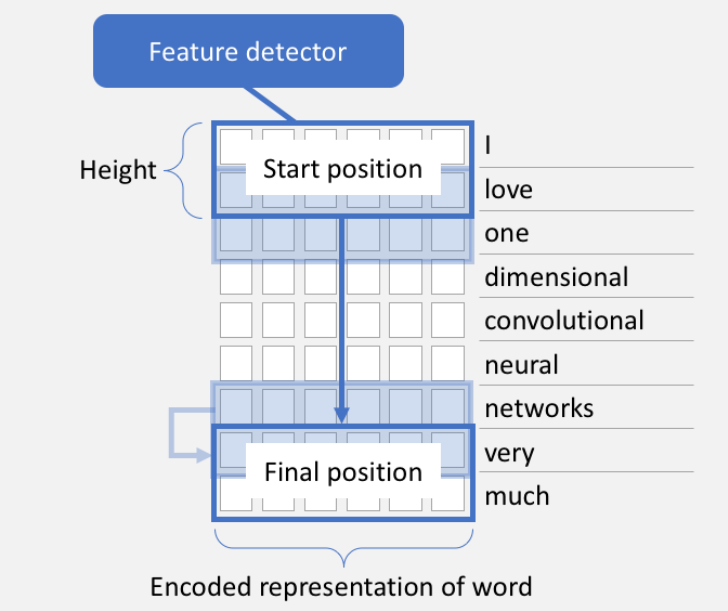
Sprint 01

Work done

The week has been spent studying and implementing the Deep Residual Convolutional Neural Network model detailed by Zhang et al.

Unlike the sequential models we proposed earlier which use 2D convolutions, the model used by Zhang el al use 1D convolutions, which scan over the d x n matrix originally proposed to produce a feature map.



As the original article was highly technical, using confounding terms and overall under-detailed, a bit of improvisation was required in implementing the model. Consequently, the model will not be a one to one translation of the original, whatever that original might have originally been.

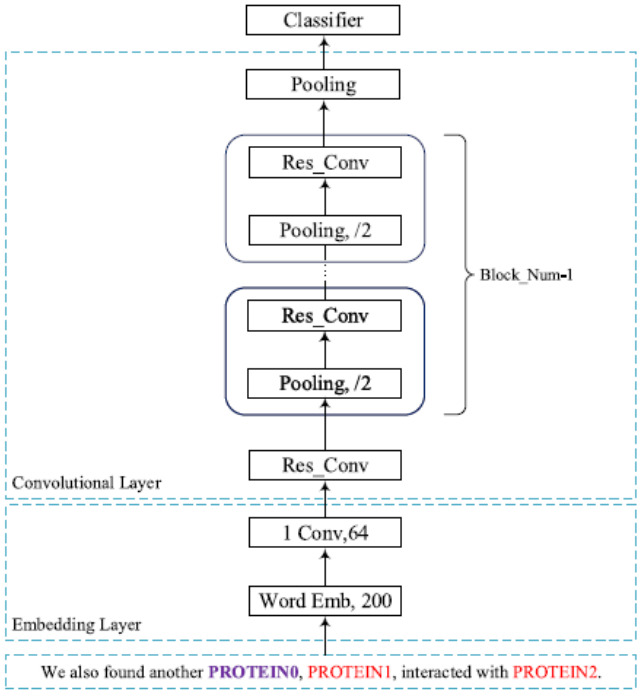
The basic building block to the Zhang model consists of a convolutional layer of 3 filters with a size of 64 and a stride of 1. This is followed by a batch normalization layer. Several of these paired layers are placed after one another in what we call the a convolutional block(conv_block).

The main unit of the mode, however, consists of the results of one of the convolutional blocks above in vector addition to the original output. This is known as a residual convolutional block(res_conv), with each one preceeded by a 1D maxpooling layer.

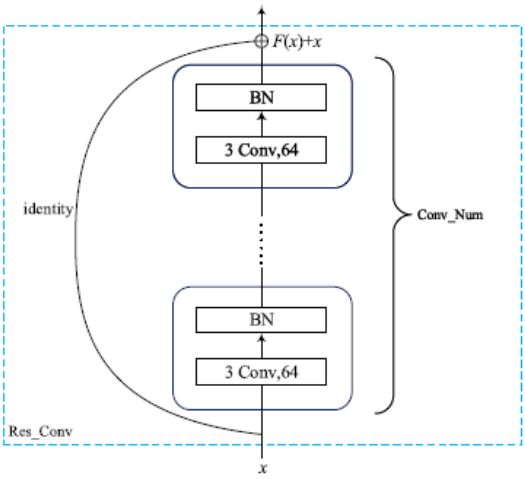
Next is the manipulation of input and output. While the original article mentioned using word vectors of size 200, we chose to maintain our initial glove embedding of 100 for ease of implementation. The model is flexible enough to allow modification in the future, which will surely arise when doing further modifications of the embedding layer.

After the embedding layer which we've set to trainable, we obtain a 100 x 100 result which is then passed onto an initial convolution of 1 filter, of window size 64 and stride 1. Then we follow up with a set of 6 residual convolutions(6 res_conv), each containing 3 convolutional blocks(3 conv_block). These unit numbers are hyperparameters that have been tested extensively in the original article, as such we will leave these untouched for now.

All that remains is the ouput. We pass the result of the deep leaning model to a global pooling layer which flattens the result of all the filters and then a dense layer for our 0 or 1 output. Below is a visual summary of the architecture, which together with our description, should provide a clearer picture.



(a) Our Model



(b) Residual Convolutional Block

Results

Results are encouraging. Our first training for 200 epochs on 11492 examples has produced an accuracy of 99.060214. When tested on the 2538 examples in our test set, however, leaves much to be desired: 60.677701.

It's clear the network overfits, which brings us to techniques to prevent this. Our first method would be data augmentation, but since it is not possible at this time to collect more scientific data nor is it possible to "skew" text, we must rely on other methods.

The dropout method involves randomly dropping a set of parameters in the hidden layers. This way, the network does not over rely on any particular set, pushing it towards generalization. We implement another network with a 0.2 dropout for each convolution, obtaining Training Accuracy: 97.955102 and Test Accuracy: 55.516154. No improvement has been observed.

Another method at our disposal is regularization. In simple terms, this involves adding "weights" to the loss function to stabilize a model regression towards linearity, preventing an overly non-linear model that fits the data too well. We combine an L2(Ridge Regularization) at 0.02 with the dropout in the previous network, and train for 200 epochs. Results are better. Although the Training Accuracy is lower at 84.319526, our Test Accuracy rises to 72.813237.

Problems

Issues remain:

1. It is unclear if our interpretation of the Zhang article is a correct one. Although it seems to produce results, perhaps a 2D convolution approach could be attempted again.
2. Since the model was modified with 100 sized word vectors, perhaps the unit hyperparameters are no longer optimal and should be tested.
3. The overfit remains. Our third attempt shows promise, and we should build on it.
4. The embedding, although set to trainable, likely still can't handle a large part of the vocabulary.

Work to do

The next phase is likely to be a more in-depth study of the dataset, as well as attempting other embedding weiths such as fasttext or scispacy.