Movie Review Sentiment Analysis

1. Introduction:

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations.

The motive of this project is to understand movie rating based on the review people have provided in texts.

Most sentiment prediction systems work just by looking at words in isolation, giving positive points for positive words and negative points for negative words and then summing up these points. That way, the order of words is ignored and important information is lost.
Let's look at the below example statement:
"This movie was actually neither that funny, nor super witty"
A naive sentiment analysis tool that predicts based on individual words may consider the above statement as positive since it has 2 positive words **"funny"** and **"witty"** but actually the entire statement is negative. In contrast, our new deep learning model actually builds up a representation of whole sentences based on the sentence structure. It computes the sentiment based on how words compose the meaning of longer phrases. This way, the model is not as easily fooled as naive models. For example, our model learned that funny and witty are positive but the complete sentence is still negative overall.

The expected output is to label phrases on a scale of five values: negative, somewhat negative, neutral, somewhat positive, positive. Obstacles like sentence negation, sarcasm, terseness, language ambiguity, and many others make this task very challenging.

2. Dataset

In this study we'll use the the Rotten Tomatoes dataset. The dataset is comprised of tab-separated files with phrases. The train/test split has been preserved for the purposes of benchmarking, but the sentences have been shuffled from their original order. Each Sentence has been parsed into many phrases by the Stanford parser. Each phrase has a PhraseId. Each sentence has a SentenceId. Phrases that are repeated (such as short/common words) are only included once in the data.

- **train.tsv** contains the phrases and their associated sentiment labels. We have additionally provided a SentenceId so that we can track which phrases belong to a single sentence.
- **test.tsv** contains just phrases. You must assign a sentiment label to each phrase.

Few examples from the training set is as below:

PhraseId	SentenceId	Phrase	Sentiment (label)
16	1	that what is good for the goose	2
17	1	that	2
22	1	good for the goose	3
34	1	the gander , some of which occasionally amuses but none of which amounts to much of a story	1

In the above example we can see how the same sentence **"A series of escapades demonstrating the adage that what is good for the goose is also good for the gander , some of which occasionally amuses but none of which amounts to much of a story"** have different sentiment when used in parts.

The sentiment labels are:

0 - negative

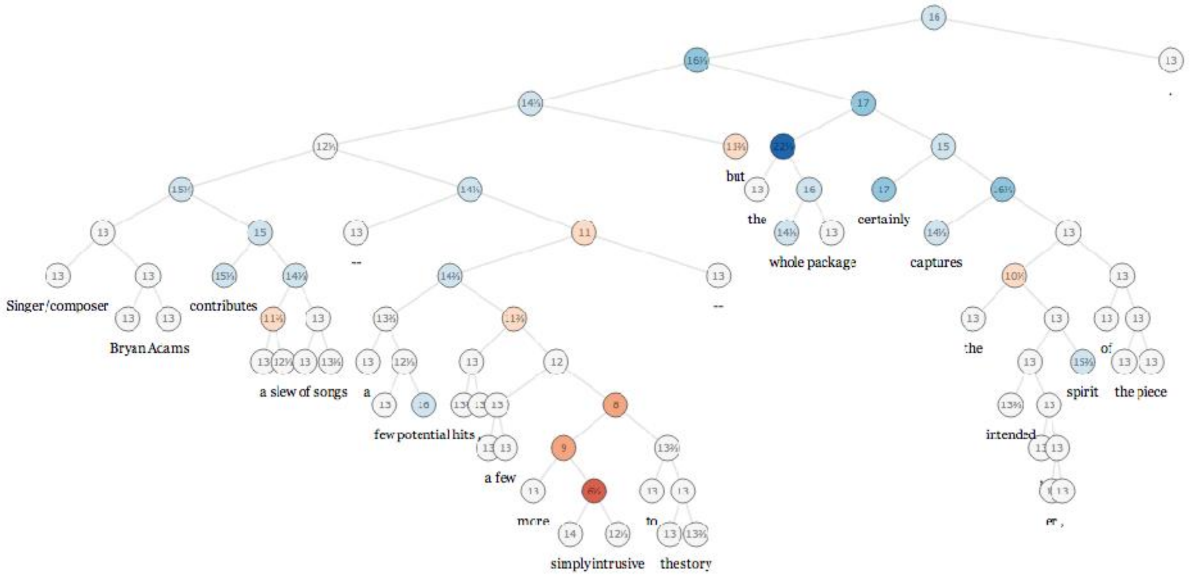
1 - somewhat negative

2 - neutral

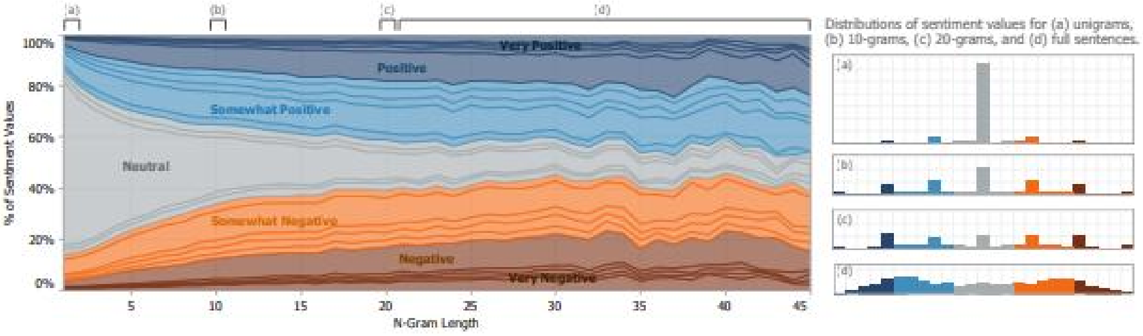
3 - somewhat positive

4 - positive

Below is one example of treebank:



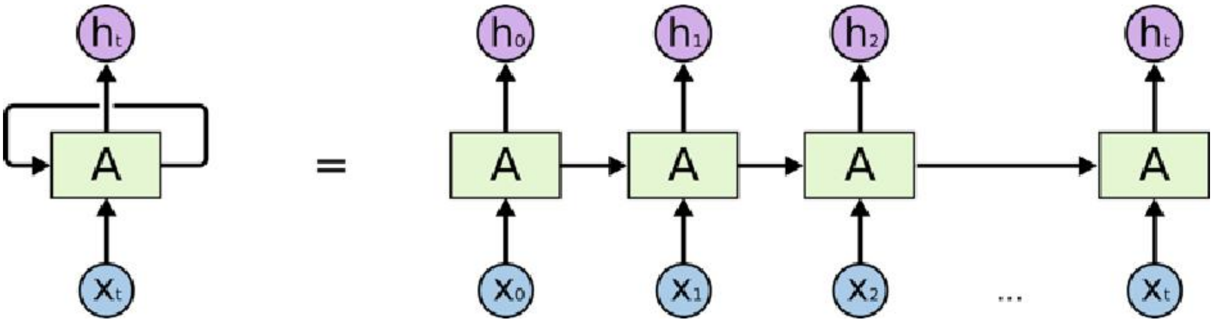
The original dataset includes 10,662 sentences, half of which were considered positive and the other half negative. Each label is extracted from a longer movie review and reflects the writer's overall intention for this review.



Above figure shows the normalized label distributions at each n-gram length. Starting at length 20, the majority are full sentences. One of the findings from labeling sentences based on reader's perception is that many of them could be considered neutral. We also notice that stronger sentiment often builds up in longer phrases and the majority of the shorter phrases are neutral. Another observation is that most annotators moved the slider to one of the five positions: negative, somewhat negative, neutral, positive or somewhat positive. The extreme values were rarely used and the slider was not often left in between the ticks. Hence, even a 5-class classification into these categories captures the main variability of the labels

3. Solution Statement

As stated in the problem statement, we need to extract the meaning of the entire statement. For that purpose, our neural network must consider events from previous steps. For this purpose Recursive Neural Network is good choice.

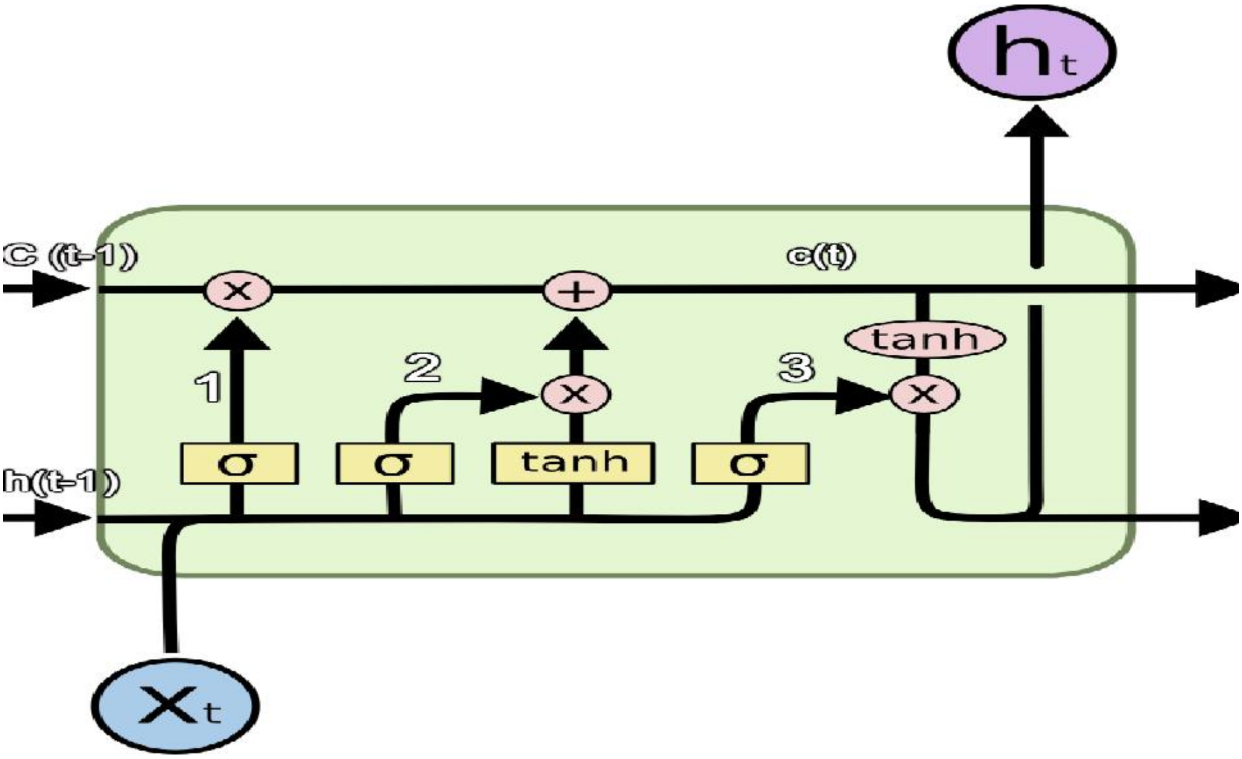


A typical RNN looks like above-where X(t) is input, h(t) is output and A is the neural network which gains information from the previous step in a loop. The output of one unit goes into the next one and the information is passed.

During the training of RNN, as the information goes in loop again and again which results in very large updates to neural network model weights. This is due to the accumulation of error gradients during an update and hence, results in an unstable network. At an extreme, the values of weights can become so large as to overflow and result in NaN values.

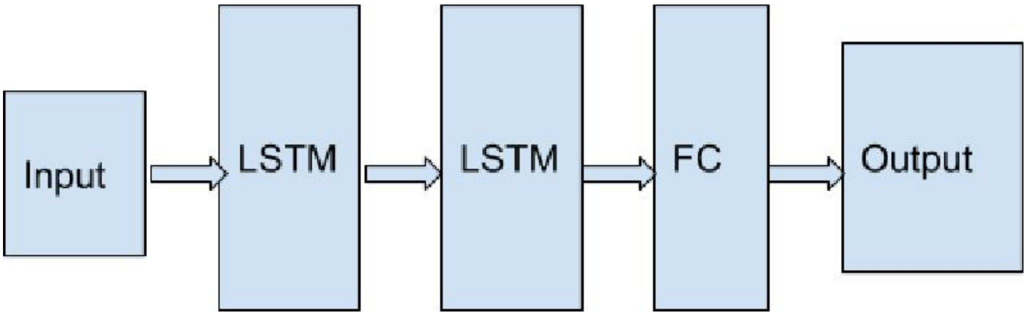
The explosion occurs through exponential growth by repeatedly multiplying gradients through the network layers that have values larger than 1 or vanishing occurs if the values are less than 1.

To overcome this problem, we'll use a variant of RNN called Long Short Term Memory(LSTM). LSTM uses gates to control the memorizing process.



To overcome the vanishing gradient problem, we need a function whose second derivative can sustain for a long range before going to zero. tanh is a suitable function with the above property. As Sigmoid can output 0 or 1, it can be used to forget or remember the information.

The neural network architecture to train the model will be as follows. We'll have 2 LSTM connected to fully connected layer which finally outputs one of the 5 categories of sentiments.



5. Evaluation Metrics

We have used mean square error (MSE) to observe the performance of the simple CNN estimator during training. At each step of the training, we will calculate MSE.

7. References:

https://www.kaggle.com/c/movie-review-sentiment-analysis-kernels-only
https://nlp.stanford.edu/sentiment/
https://towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-for-sentiment-analysis-af410fd85b47