Emotion Recognition based on Lyrics

Aishwarya Ramachandrappa
Jinhao Wei

Abstract*:* The music industry, being a billion-dollar success, has been giving huge importance to the advancement of technology. It finds its application in classification and recommendation systems. Music has and will always be an important factor in our lives. Sentiment analysis of text content is important for many natural language processing tasks. Analyzing the sentiment of any song has various applications such as mood-based playlist creation, personalized recommendation system etc. In this project, we used the textual features of the song to classify them by mood. Our model consists of two parts: the word embedding and the neural network. The word embeddings are used to convert each word into a vector representation. Thus, in later steps, we will be able to use the vectors, instead a non-recognizable word, as inputs to our neural network. We had experimented on serveral models, but we would like to focus our introduction on two of them. The first model is the word2vec + CNN and the second model is the FastText + LSTM network. The CNN model is more accurate than similar shallow convolution networks or deeper recurrent networks.

Index Terms: Deep Learning, Neural Network, Sentiment Analysis, Word2Vec, FastText, CNN, LSTM, MER

I. INTRODUCTION

Neural network and sentiment analysis are two of the most discussed topics that are under research and constant evolution in the recent years. Neural networks, which are modeled after the human brain, are algorithms that are designed for pattern recognition. The patterns to be recognized are usually numerical objects contained in vectors. Theoritically, all real-world problems can be solved by pattern recognition (imagine the case that our world is formed up by countless of 0 and 1). Neural networks help us cluster and classify. They act as a layer above the dataset which helps in the classification and recognition of the dataset to find the corresponding label.

Sentiment analysis is the process of understanding the intention or emotion behind a given piece of text. The classification is done based on emotion such as joy, anger, relaxed, sad, etc. It is performed to understand the general opinion across many documents within a corpus. One of the applications of sentiment analysis is Music Emotion Recognition. Every song is created to depict some emotion, which can be interpreted either with the lyrics or the audio features. Since lyrics are much easier to be obtained and classified (compared to the audio features), we decide to use lyric texts to classify songs based on the mood they are trying to convey. Experimental results show that textual features outperform audio-only classifiers on most classification tasks.

Our dataset consists of 4 moods – happy, sad, angry and relaxed. The corpus is first passed to the word embeddings models to get the vector representation of every word(There are already several word embedding methods, such as GloVe, Word2Vec, FastText, etc.). In the next step, the result of the embeddings will be fed to our neural network, such as N-gram CNN and LSTM. As for the results, the NgramCNN gives the best accuracy as it considers various subsequence of the entire lyrics, thus keeping the semantics of the lyrics to understand the content. The LSTM, however, gave an accuracy that is sightly worse than NgramCNN.

II. DESIGN

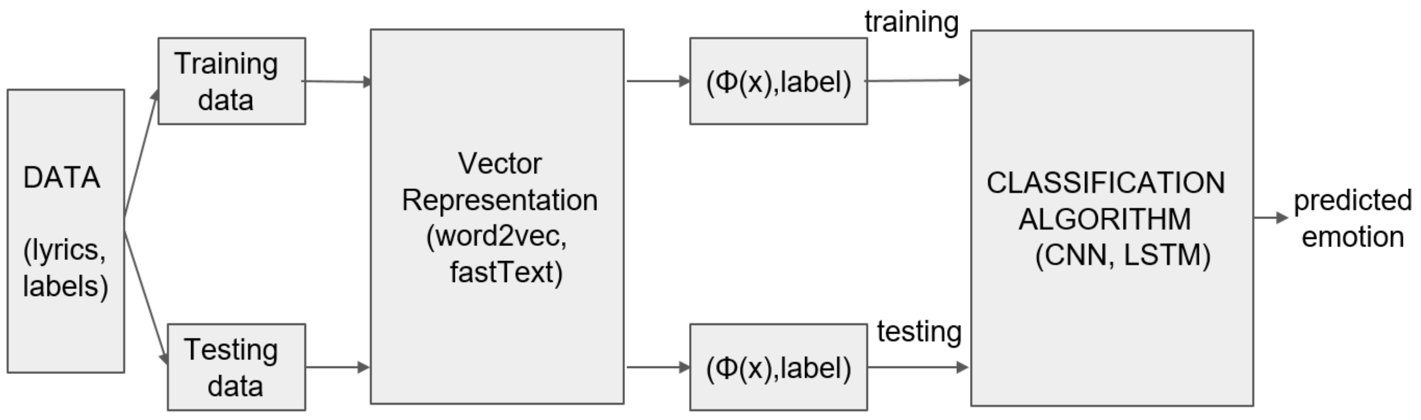


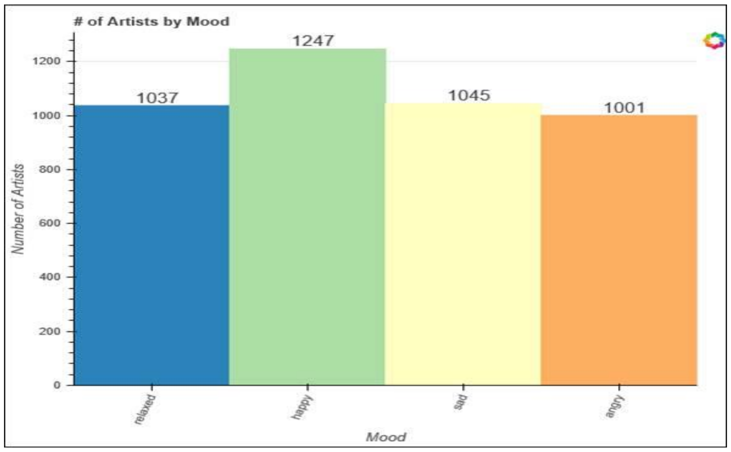
Figure 1: Proposed Model

A. Dataset

Due to the evolving music industry, there are several datasets available. The motivation behind the dataset was obtained from the MIREX [1], which was created for the purpose of emotion recognition. But the disadvantage of this dataset is its limited number of songs and large amount of labels. There were 681 songs classified into 28 classes in this dataset, which makes it very difficult to train the neural network. There are also some other dataset with larger scale and fewer labels, but most of them are categorized by genre. This is not suitable for our deep learning task, hence the dataset had to be manually created.

Two existing datasets, MoodyLyrics and MoodyLyrics4Q are used to get the mood of the songs. But the lyrics of the songs were removed due to copyright issue. Thus, the dataset was created by scrapping the lyrics from lyrics.wikia.com. The MoodyLyrics dataset is based on the Circumplex model of emotions. This model has 4 basic moods – happy, angry, sad and relaxed, based on the valence and arousal value.

Statistics	Value
Number of songs	4330
Number of words	969924
Number of unique vectors	27485
Maximum length of the song	1162
Number of Artists	2548



B. Word Embeddings

1. **Word2Vec**: Each word from the dataset has a unique vector representation. The words with common contexts in the corpus are in close proximity with each other in the vector space. There are 2 main learning algorithms:

1.1 Continuous bag-of-words (Cbow): During the training of Cbow model, we will give the context of the corpus and learn to predict the middle word. For example, the input to the model could be w_{i-2}, w_{i-1}, w_{i-1}, w_{i+2}. Our goal will be to train the model and finally output the word w_i. This training process means that we consider the words that are before and after the current words. Thus, this is known as the "predicting the word given its context" method. The number of words we use in the training process depends on the window size argument, which can be manually set. All the words within the window size are taken into accounts and the relationship between the context words and the target word is mapped and finally predicted.

1.2 Skip-gram: The input to the model is w_i, and the output could be w_{i-2}, w_{i-1}, w_{i-1}, w_{i+2}. This is the "predicting the context given a word". As the distance of the words increase, they are given less weights by randomly sampling them. In this case, there is no sliding window and the maximum window size is set which considers all the words in the data. The actual size of the window is selected randomly between 1 and the maximum window size.

2. **FastText:** FastText's unsupervised mode contains both the Cbow and Skip-gram model. In the meantime, it used another technique named "n-grams", which means that it does not only focus on each word, also, every sub-word will also be taken into accounts. For example, while training the word "where", it will be divided into sub-words "whe", "her", "ere". Finally, the word "where" will be represented as the sum of all the vector of its sub-words. This was meant to make sure the words that has the same prefix or suffix could be closer in the vector space, which could possibly lower the deviations.

C. Deep Learning Models

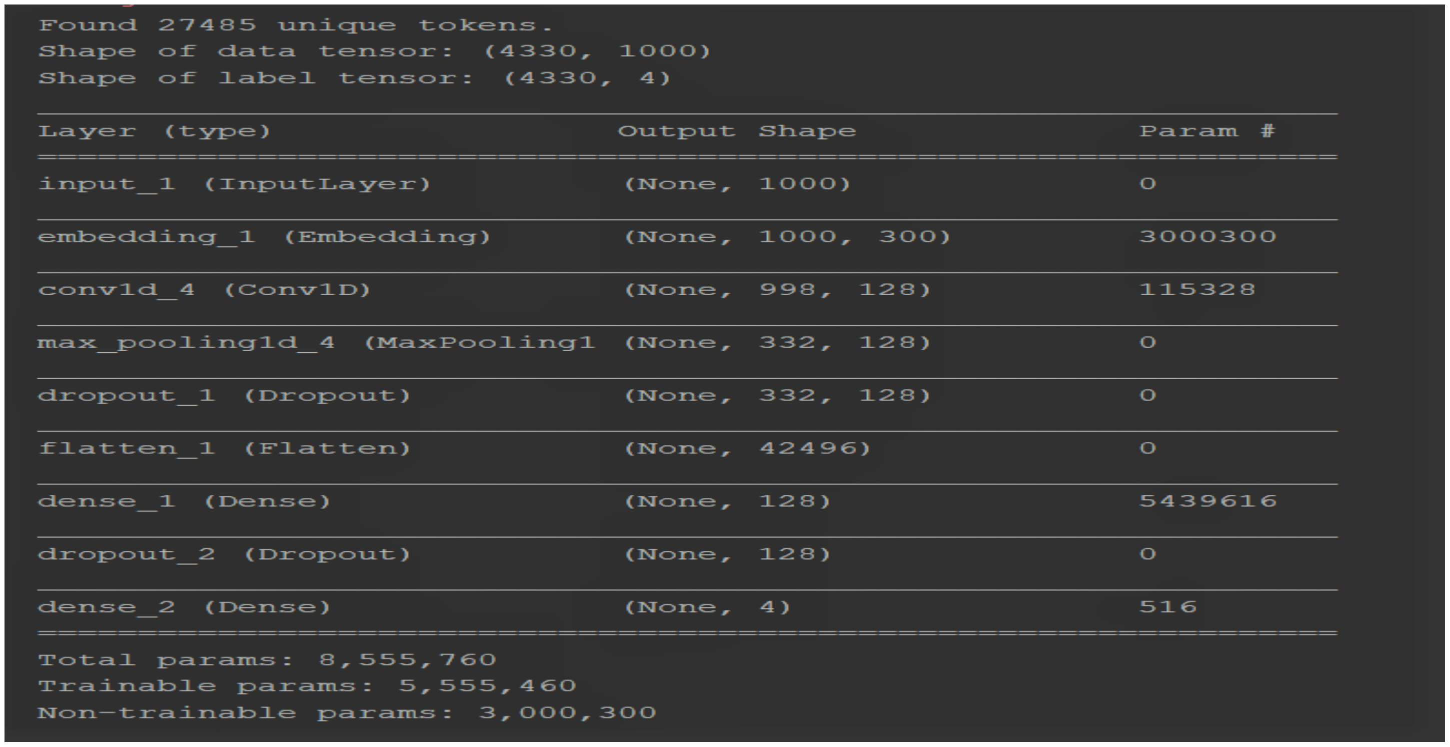
1. **CNN:** CNNs are known to perform well on data with high locality, when words or features care more about the features surrounding them. CNNs have the benefit of being more computationally efficient than LSTMs, which enabled us to feasibly test character-level embeddings in addition to word embeddings. Yoon Kim's CNN model [3] for sentence classification is used as it considers the sub words taking multiple n-grams. This helps in understanding the meaning of the lyrics better.

2. **LSTM:** LSTMs outperform traditional RNNs in most cases, given that it was implemented with GRU cells. On one hand, the recurrent propertity of GRU cells allow LSTM network to keep a long-term memory. On the other hand, the GRU cells successfully prevent the vanishing gradient problem in the network. Therefore, LSTMs are commonly used in text sentiment analysis, as it can fully memorize and interpret the text from begin to the end.

III. MODELS

1. **Word2Vec + CNN*:*** The first model we would like to present is the word2vec data embedding which contains feature vectors in 300 dimensions for about 3 million words and phrases, extracted from Google News. the words that are not in the word2vec vocabulary U [−a, a] where a is chosen such that the unknown words have the same variance as words already in word2vec[2]. The lyrics are then padded to make the dimensions of all the songs the same. Each song now has 1000-word sequence. This is now passed to the convolutional neural network. The first layer is 1D- Convolutional layer with ReLU. Multiple feature maps and filter widths are used. Filter widths of 3, 4, 5 are the corresponding gram thus splitting the sequence and understanding it in more than one way. Max-pool the result of the convolutional layer into a long feature vector. There is a drop out of 0.5 which is then passed to the dense layer with 128 units. The model gives us a test accuracy of 82% thus classifying the moods efficiently.

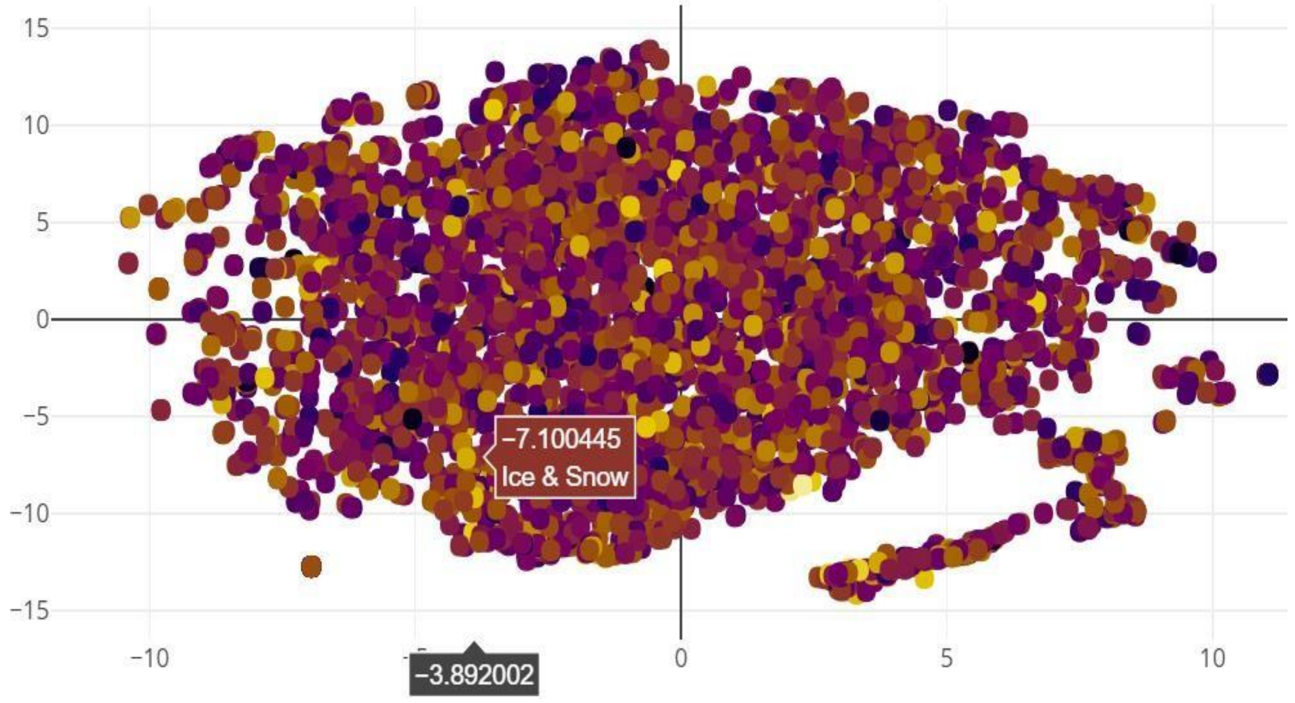
2. **FastText + LSTM:** In this model, we used Fasttext's cbow algorithm to pre-train our word vectors. In later steps, we applied these word vectors to our uni-directional LSTM network and trained our model. For the word vectors, each word was trained into a vector of length 100. For each piece of lyric, only the first 100 words are used for classification. The hidden units in each LSTM cell was set to be 128. In the meantime, there was no mini-batch and drop-out method applied in this model. This model finally gave an accuracy of approiximately 75% at around epoch 90.



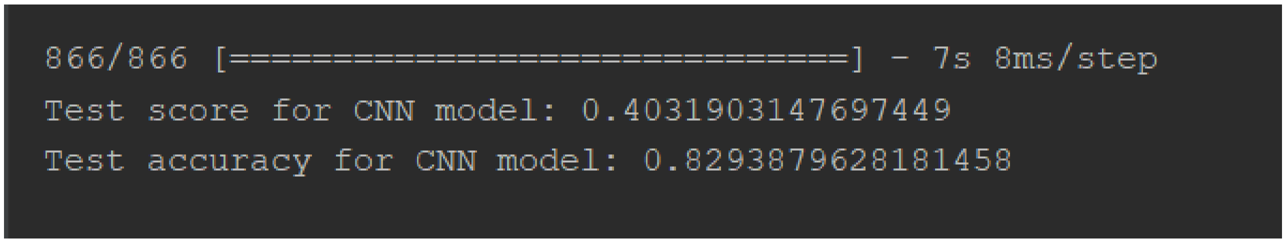
```
Found 27485 unique tokens.
Shape of data tensor: (4330, 1000)
Shape of label tensor: (4330, 4)
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1000)              0
_________________________________________________________________
embedding_1 (Embedding)      (None, 1000, 300)         3000300
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 998, 128)          115328
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 332, 128)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 332, 128)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 42496)             0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               5439616
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 516
=================================================================
Total params: 8,555,760
Trainable params: 5,555,460
Non-trainable params: 3,000,300
```

IV. RESULTS

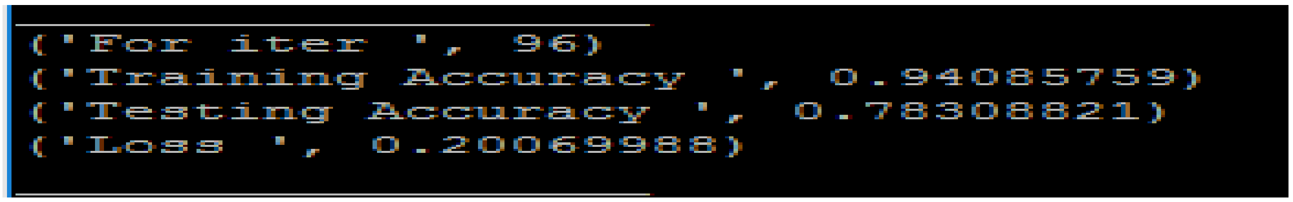
A. **Word2Vec representation**: For the better understating of the Word2Vec model, a scatter plot is plotted to show the vectors of every lyric. The is obtained by finding the normalized sum of the word vectors. This helps us understand the songs that are similar and thus make the classification easier.



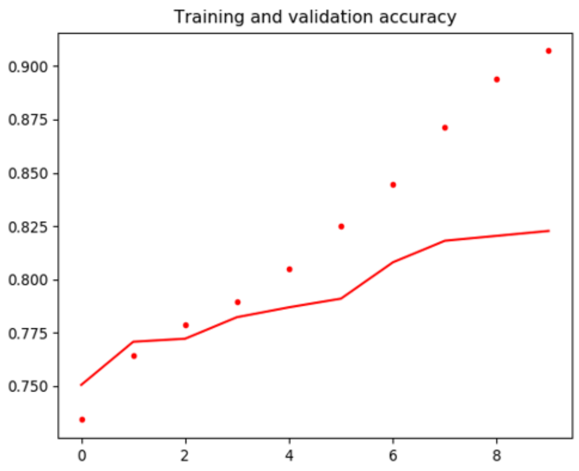
B. Accuracy: The following graphs show the comparison of the two models used in the project. The training and validation data is plotted against epoch.



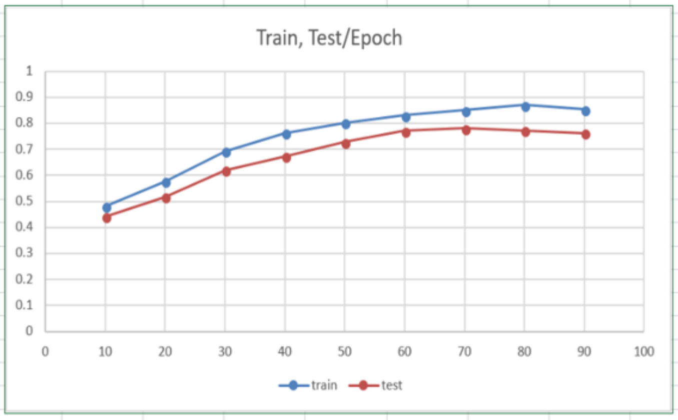
```
866/866 [==============================] - 7s 8ms/step
Test score for CNN model: 0.4031903147697449
Test accuracy for CNN model: 0.8293879628181458
```



```
('For iter ', 96)
('Training Accuracy ', 0.94085759)
('Testing Accuracy ', 0.78308821)
('Loss ', 0.20069988)
```

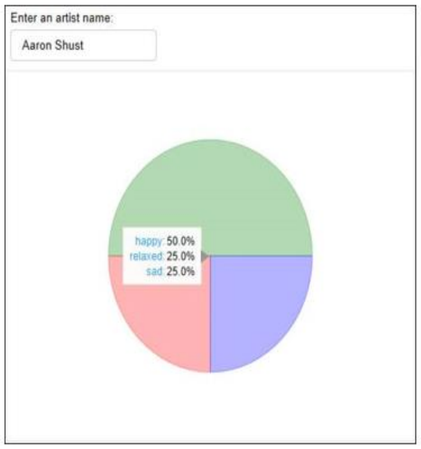


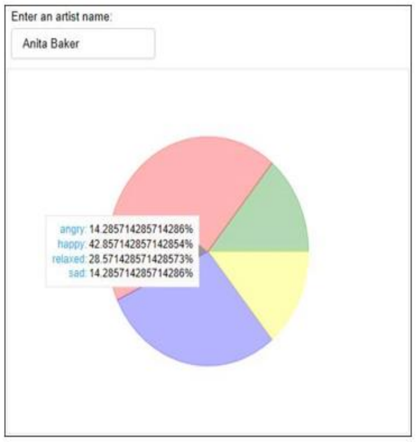
Word2Vec - CNN



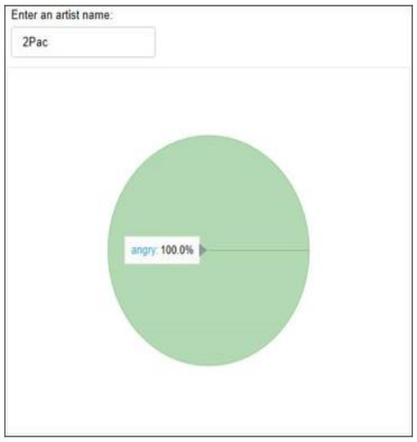
FastText - LSTM

C. Artist-Mood relation: To understand the type of songs written by a particular artist, a pie chart is plotted. This helps is understanding the dataset better and shows the details of the mood an artist is targeting.





Enter an artist name:
Anita Baker
angry: 14.285714285714286%
happy: 42.857142857142854%
relaxed: 28.571428571428573%
sad: 14.285714285714286%



V. FUTURE SCOPE

The project implemented as such is a small subset of all that is possible with sentiment analysis. The scope of immediate future work possible for this project would be:

1. Dataset Preprocessing:
   - Removal of songs which were from certain non-English artists which were identified
   - Remove non-ASCII character and non-whitespace character
2. Bucketing can be implemented instead of padding which helps obtain better accuracy.

REFERENCES

1. Luis Cardoso, Renato Panda, Rui Paiva, "MOODetector: A Prototype Software Tool for Mood-based Playlist Generation"
2. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Efficient Estimation of Word Representationns in Vector Space" arXiv: 1301.3781
3. Kim, Yoon." Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).
4. Theodora Chu, Kylie Jue, Max Wang, 'Comment Abuse Classification with Deep Learning'
5. Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, "Bags of Tricks for Efiicient Text Classification" arXiv 1607.01759
6. https://jasdeep06.github.io/posts/Understanding-LSTM-in-Tensorflow-MNIST/
7. http://colah.github.io/posts/2015-08-Understanding-LSTMs/