**Fine-Grained**

**Sentiment Analysis on Financial Microblogs**

**and News**

**Task 1: Sentiment Analysis of Microblogs:**

* **Dataset :**

Training Set – 1204

Test Set – 335

Validation Set -134

**Approach 1 Using Gensim WordVectors:**

**Pre-Processing:**

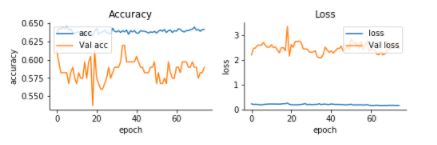
* Removal of special characters, punctuations and numbers.
* Removal of URLs, user names mentioned in a tweet message.
* Removal of words with length less than three in order to reduce the dimensionality of feature space.
* Conversion of tweet text into lower case.
* Concatenation of spans to form a unified string. For the empty spans field, we considered the whole preprocessed message text for feature extraction.

**Features:**

* Word2Vec genism model built using [Finance model (all\_fin\_model\_lower)](https://github.com/apmoore1/semeval/blob/master/models/word2vec_models)  which had collection of 189,206 financial articles.
* Average Weighted vectors built by combination of above word2vec model and tf-idf vector calculated from the dataset.
* The span column is the only text data fed as input.

**Model 1 MLP with unaltered sentiment values:**

* Sequential layer followed by dense layers and dropout of 0.2 and 0.5 respectively.
* Tanh activation function is used .
* RmsProp optimizer is used, with binary\_cross\_entropy as loss function.
* epochs=75, batch\_size=120

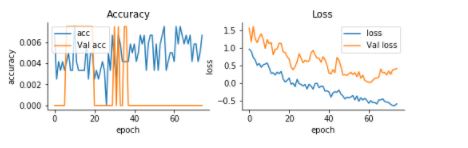




* Accuracy : Less than 1%
* **Cosine\_Similarity: 0.46058743**

**Model 1 MLP with Sentiment scores normalized to 0’s and 1’s:**

* 



* Accuracy: 55.2238812198
* **Cosine\_Similarity:** 0.55062247

**Approach 2 Using Lexicon Features:**

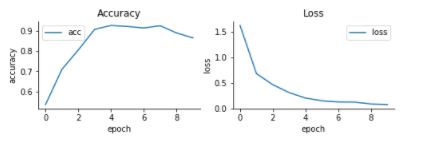
**Pre-Processing:**

* Positive and negative word count is calculated using Loughran and McDonald Sentiment Word Lists
* For Each POS , positive and negative sentiment score is determined using stock market lexicon
* TF-IDF is calculated for the text and data is transformed into vectors.
* All the newly created columns are merged together and finally split into training and test data.

**Model 1 MLP with Sentiment scores normalized to 0’s and 1’s:**

* The model is built with same design followed in the genism model.





* Accuracy : 0.5880597052289479
* **Cosine\_Similarity: 0.81846746**

**Model 2 CNN with Sentiment scores normalized to 0’s and 1’s:**

* Convolutional 1-D layer is added as input layer and tanh activation is used as the activation function.
* Convolutional layers with sigmoid activation is added next
* Dropout layers are added in between the conv layers.
* Then model is flattened with flattening layer
* Dense layer with tanh activation is added as output layer.
* Adam optimizer is used along with cosine\_proximity loss function.
* 15 epochs are run with batch size of 128.
* Accuracy: 0.6507462666995489
* **Cosine\_Similarity: 0.80668845**

**Model 3 LSTM with Sentiment scores normalized to 0’s and 1’s:**

* Sequential model is built with LSTM layer as input.
* We are having 128 LSTM hidden units.
* The dense layer with one output is added as output layer with tanh activation function.
* Adam optimizer is used along with cosine\_proximity loss function.

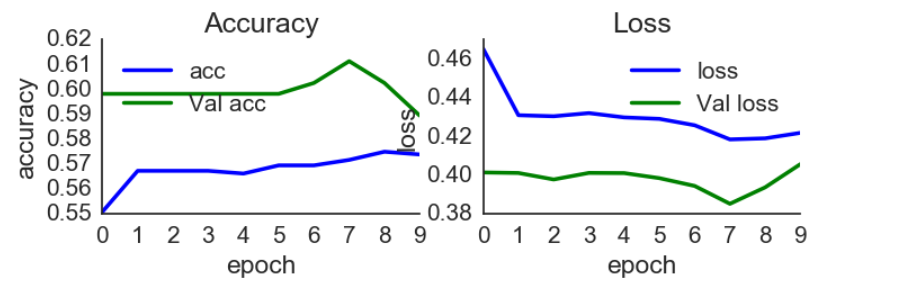
**Task 2: Sentiment Analysis of News Data:**

**Pre-Processing and Features:**

* Glove 100 dimensional vector data is used for embedding layer creation.
* The text data is tokenized and converted into sequence using keras tokenizer and text\_to-Sequence function.
* The padding is done obtaining the maximum length of each sentence using pad\_sequences function.
* Data is divided into training and test data with test size of 0.2.
* Pre-trained word embeddings is used to build the embedding layer along with the word index.

**Model 1: MLP:**

* The embedding layer is fed as input to the sequential model after reshaping the training data.
* The flattening layer is added to flatten the embedding sequence.
* Dense layer with 256 hidden units added followed by dropouts
* The output dense layer is added with tanh activation function.
* Adam optimizer is used along with cosine\_proximity loss function.
* 10 epochs are run with batch size of 64
* Model is fit with Accuracy: 59.83%
* **Cosine\_Similarity: 0.668845**



**Model 2 CNN:**

* Embedding layer is added as input layer and tanh activation is used as the activation function.
* Convolutional 1-D layers with linear activation is added next
* Dropout layers are added in between the conv layers.
* Then model is flattened with flattening layer
* Dense layer with tanh activation is added as output layer.
* Adam optimizer is used along with cosine\_proximity loss function.
* 20 epochs are run with batch size of 64.
* Accuracy: 0.60
* **Cosine\_Similarity: 0.77342538**

