Sentiment Analysis on Financial Data Using Neural Networks

Approach 1

We have combined hand-engineered lexical, sentiment, and metadata features with the representations learned from deep-learning approaches like Convolutional Neural Networks (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU).

Analysis of Data

Predicting sentiments of financial data has diverse applications. Positive news has the capability to boost the market by increasing optimism among people. 'Fine-Grained Sentiment Analysis on Financial Microblogs and News' aims at analyzing the polarity of public sentiments from financial data present in newspapers and social media.

Data Distribution

Task	Training	Trial	Test
Subtask-1	1,694	10	799
Subtask-2	1,142	14	491

The dataset is noisy and contains URLs, cashtags, digits, usernames and emoticons. The messages are short with an average number of 13 tokens for the microblog data and 10 tokens for the headlines data.

Feature Engineering

Before extracting the features, the following steps are applied as a part of Data
(a)Converting the text to lowercase
(b)Stemming
(c)Replacing named entities(NE) and digits with common identifiers.
The above-mentioned steps are performed to reduce noise in the data.

Lexical:
We extracted word n-grams (n=1,2,3) and character n-grams (n=3,4,5) from the messages.

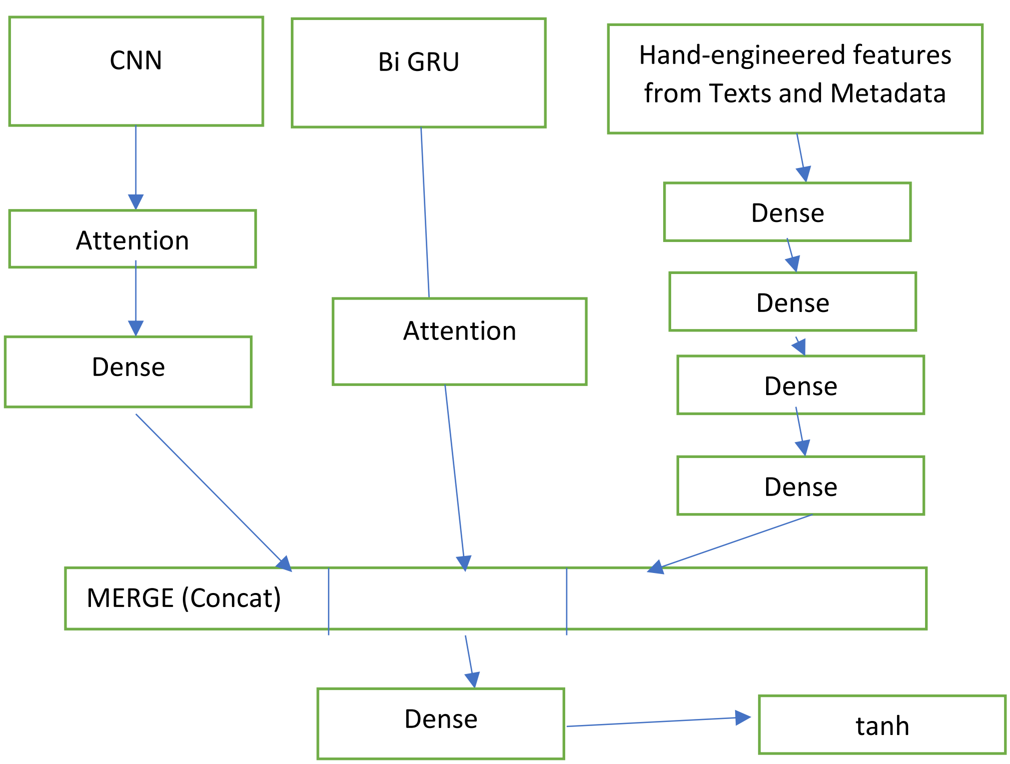
Sentiment:
We used SenticNet library as it provides a collection of concept-level opinion lexicons with scores in five dimensions (aptitude, attention, pleasantness, polarity, and sensitivity). Both stemmed and non-

stemmed versions of the messages are used to extract concepts from the knowledge base. Concepts are modeled as bag-of-concepts(BOC) and used as binary features.

cashtag	id	sentiment :	source	spans	concepts	polarity	attention	pleasantr	aptitude	sensitivity	stemmed_pol	stemmed_atten	stemmed_pleas	stemmed_aptitu	stemmed_sensi	stemmed_concept
$FB	7.20E+17	0.366	twitter	consumers	cautious stance ke	0.177333	0.389667	-0.1	0.419	-0.173333	0.136	0.228	0.02	0.306	-0.14	keep
$LUV	7.20E+17	0.638	twitter	NAMEDEN	entry	0.073	0.066	0	0.154	0	0	0	0	0	0	
$NFLX	5329774	-0.494	stocktwits	Every Reas	every reason	-0.341	0.143	-0.35	-0.2965	0.338	0.078	0.286	-0.09	0.227	-0.18	reason
$DIA	7.20E+17	0.46	twitter	NAMEDEN	high all time need	0.19575	-0.3675	0.08	0.306	0.219	0.19575	-0.3675	0.08	0.306	0.219	high all time need
$PLUG	20091246	0.403	stocktwits	Long setup	long	0.056	0.065	0	0.142	0.039	0.056	0.065	0	0.142	0.039	long
$GMCR	5819749	0	stocktwits	will be a so	today term passive	0.166833	0.0855	0.0258	0.315	0.1628333	0.421	0.2414	0.3646	0.3518	0.0128	today term ad lon
$IBM	7.10E+17	-0.296	twitter	recall	recall	0.483	0.722	0.726	0	0	0	0	0	0	0	
$JOSB	17892972	-0.546	stocktwits	NAMEDEN	deal cash new raisi	0.451167	0.2345	0.4438	0.446	-0.0225	0.46466667	0.5175	0.4435	0.4615	-0.0225	deal cash new wir
$CSTM	7.10E+17	-0.438	twitter	NAMEDENTITY		0	0	0	0	0	0	0	0	0	0	
$PYPL	7.08E+17	0.408	twitter	NAMEDENTITY AppleStores ar		0	0	0	0	0	0.035	0	0.048	0.058	0	liquid
$GOOGL	31971935	-0.398	stocktwits	NAMEDEN	star analyst buy	0.34	-0.24667	0.492	0.212667	0	0.34	-0.246666667	0.492	0.212666667	0	star analyst buy
$ENDP	7.10E+17	-0.349	twitter	Excited big	excited value big	0.124	0.306333	0.3143	0.048	-0.263667	0.1985	0.491	0.0185	0.1115	0.0245	hope big
$XLI	13915103	0.025	stocktwits	NAMEDENTITY		0	0	0	0	0	0	0	0	0	0	
$PCLN	10448993	0.486	stocktwits	NAMEDENTITY		0	0	0	0	0	0	0	0	0	0	
$AA	24886266	0.308	stocktwits	NAMEDEN	going	-0.72	0	-0.66	-0.91	0.585	-0.55	0	-0.78	-0.88	0	go
$AAPL	12793642	-0.372	stocktwits	NAMEDENTITY NAMEDENTITY		0	0	0	0	0	0	0	0	0	0	
$AAPL	9408369	0.461	stocktwits	not guaranteed		0	0	0	0	0	0	0	0	0	0	
$GOLD	7.20E+17	0.408	twitter	Potential c	potential continua	0.312	0.562	0.3355	0.1045	-0.06	0	0	0	0	0	

Word Embeddings: Since word embeddings show semantic information, word vectors trained on Google News has been used to capture the semantic representation of the messages. It has 3M vocabulary entries. We averaged the word vectors of every word in the messages and represented them with a 300-dimensional vector. The words that are not available in the pre-trained vectors vocabulary are skipped.

Metadata:
We used the message sources, cashtags and company names as metadata features.



Cosine score=0.398

Approach 2 RNN

- **Datasets(loaddataset.py)---- Data preprocessing**
  1. Load to pandas dataframe and normalize json
  2. Get only required columns
  3. Convert span lists to text
  4. Write index in column
  5. make sentiment score numeric
- **feature engineering------**
  1. transfer word, doc to be vector
  2. pos_features, sentiment_feature, large_features, char_approach
  3. lexicon lib:
- **Model: RNN MODEL**

Grid research for finding the best paramters for model

AdamOptimizer

batch_size = 187

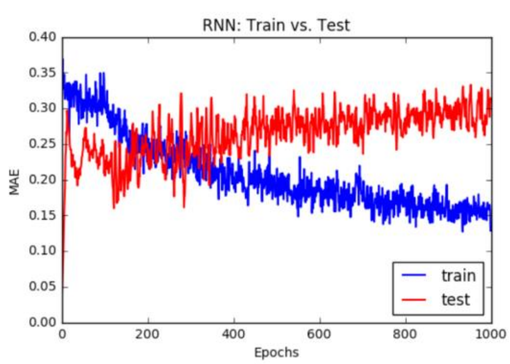
p_dropout = 0.51

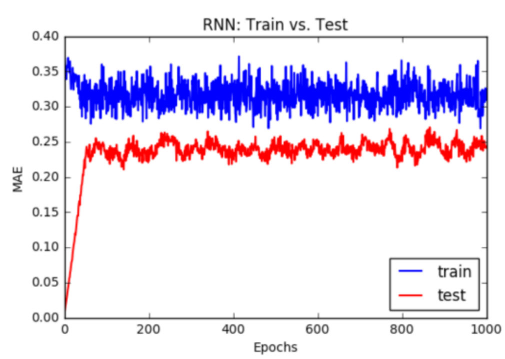
p_layer_count = 15

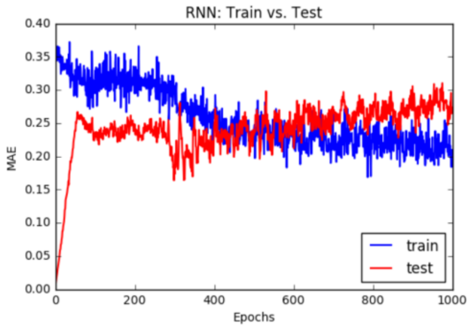
p_num_layers = 8

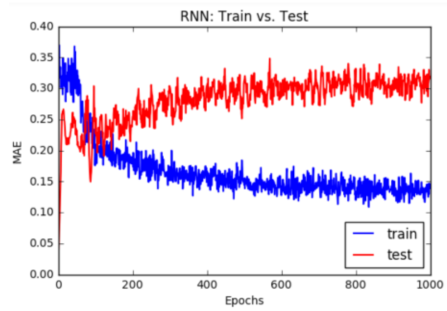
epochs = 1000

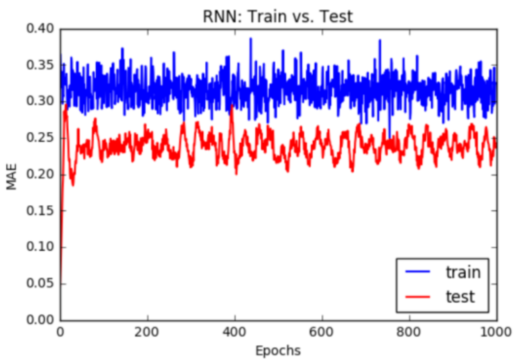
- **test vs train error for different features:**

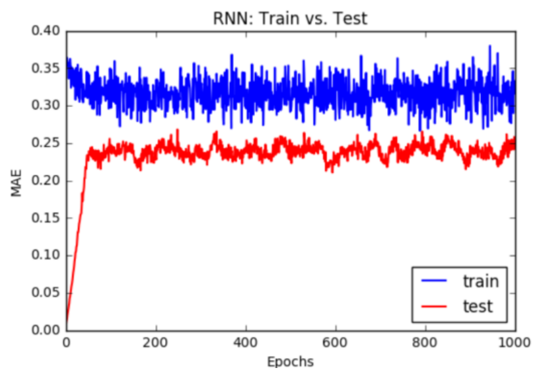








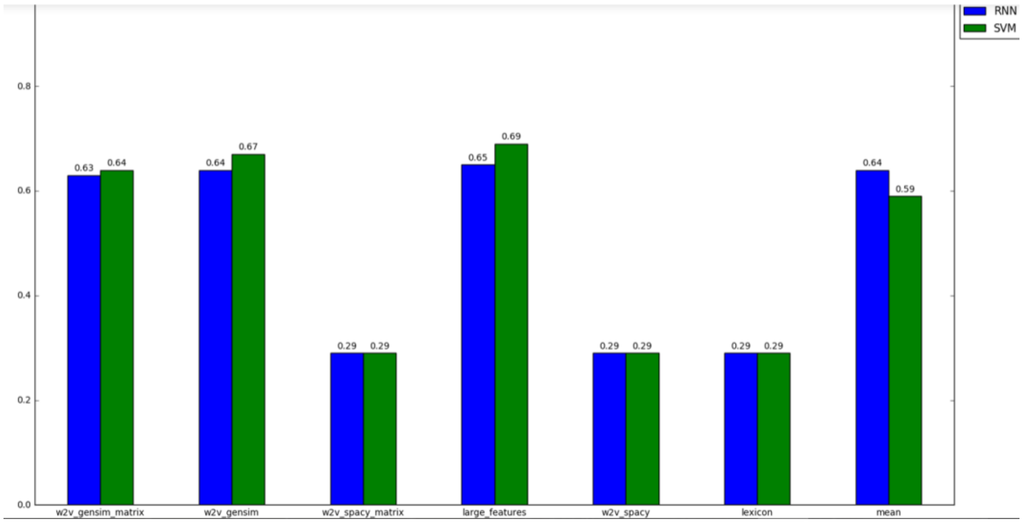




- **Cosine Similarity for each feature with tables ----- 794 rows**
  http://localhost:8888/notebooks/SemEval2017Task5-master-Tocab/main.ipynb

- **Using Kfold CV Training data cross-validation , split to 5 folder, here is the performance for each folder only using the training data to test the model :**

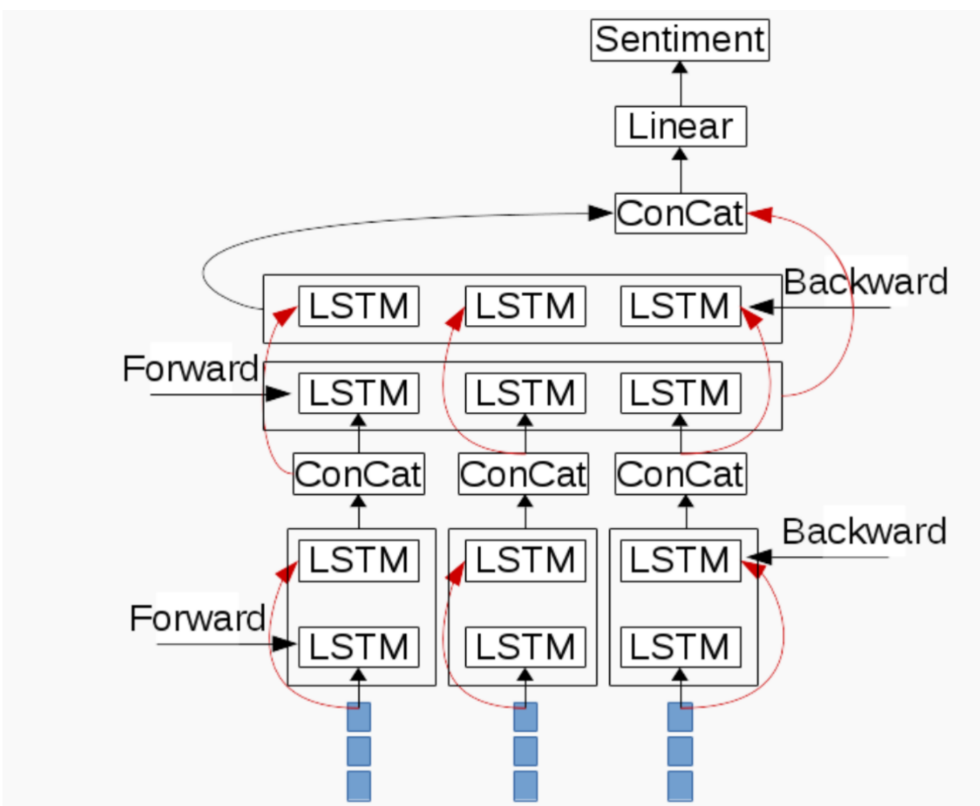
```
1. fold:SVR_w2v_gensim_matrix [[ 0.62116764]] SVR_w2v_gensim [[ 0.68102718]] SVR_w2v_spacy_matrix [[ 0.35890401]] SVR
_large_features [[ 0.68010666]] SVR_w2v_spacy [[ 0.35890401]] SVR_lexicon [[ 0.35866669]] SVR_mean [[ 0.61888199]] RN
N3_w2v_gensim_matrix [[ 0.64227225]] RNN3_w2v_gensim [[ 0.66171637]] RNN3_w2v_spacy_matrix [[ 0.35890401]] RNN3_large
_features [[ 0.66975493]] RNN3_w2v_spacy [[ 0.35890401]] RNN3_lexicon [[ 0.35890416]] RNN3_mean [[ 0.67144422]]


2. fold:SVR_w2v_gensim_matrix [[ 0.61958669]] SVR_w2v_gensim [[ 0.66647377]] SVR_w2v_spacy_matrix [[ 0.31434925]] SVR
_large_features [[ 0.66563883]] SVR_w2v_spacy [[ 0.31434925]] SVR_lexicon [[ 0.31390237]] SVR_mean [[ 0.6049875]] RNN
3_w2v_gensim_matrix [[ 0.58748905]] RNN3_w2v_gensim [[ 0.60609605]] RNN3_w2v_spacy_matrix [[ 0.31434925]] RNN3_large_
features [[ 0.60901205]] RNN3_w2v_spacy [[ 0.31434925]] RNN3_lexicon [[ 0.31435335]] RNN3_mean [[ 0.63866041]]
3. fold:SVR_w2v_gensim_matrix [[ 0.64262701]] SVR_w2v_gensim [[ 0.674]] SVR_w2v_spacy_matrix [[ 0.21010146]] SVR_larg
e_features [[ 0.70149187]] SVR_w2v_spacy [[ 0.21010146]] SVR_lexicon [[ 0.2115608]] SVR_mean [[ 0.54825379]] RNN3_w2v
_gensim_matrix [[ 0.65314751]] RNN3_w2v_gensim [[ 0.66884125]] RNN3_w2v_spacy_matrix [[ 0.21010146]] RNN3_large_featu
res [[ 0.6867044]] RNN3_w2v_spacy [[ 0.21010146]] RNN3_lexicon [[ 0.21013641]] RNN3_mean [[ 0.6180668]]
4. fold:SVR_w2v_gensim_matrix [[ 0.64546054]] SVR_w2v_gensim [[ 0.65729577]] SVR_w2v_spacy_matrix [[ 0.27646883]] SVR
_large_features [[ 0.70166514]] SVR_w2v_spacy [[ 0.27646883]] SVR_lexicon [[ 0.27602476]] SVR_mean [[ 0.58599418]] RN
N3_w2v_gensim_matrix [[ 0.61723103]] RNN3_w2v_gensim [[ 0.63800334]] RNN3_w2v_spacy_matrix [[ 0.27646883]] RNN3_large
_features [[ 0.63608327]] RNN3_w2v_spacy [[ 0.27646883]] RNN3_lexicon [[ 0.27647416]] RNN3_mean [[ 0.6445974]]
5. fold:SVR_w2v_gensim_matrix [[ 0.66833879]] SVR_w2v_gensim [[ 0.67515288]] SVR_w2v_spacy_matrix [[ 0.28344222]] SVR
_large_features [[ 0.70480254]] SVR_w2v_spacy [[ 0.28344222]] SVR_lexicon [[ 0.28376079]] SVR_mean [[ 0.597269]] RNN3
_w2v_gensim_matrix [[ 0.65521527]] RNN3_w2v_gensim [[ 0.62193237]] RNN3_w2v_spacy_matrix [[ 0.28344222]] RNN3_large_f
eatures [[ 0.64804446]] RNN3_w2v_spacy [[ 0.28344222]] RNN3_lexicon [[ 0.28344368]] RNN3_mean [[ 0.63894646]]
```



<u>Approach 3(For Headlines dataset)</u>

Preprocessing and Additional data used

1.Lower cased 2.Tokenised



Word2Vec – This was created using Gensim

Standard Model(SLSTM)

- Drop out between layers and connections
- 25 times trained over the data (epoch of 25)

Early Stopping LSTM(ELSTM)

- Drop out between layers only
- Early stopping used to determine the epoch.

Cosine Score of SLSTM=0.41

Cosine Score of=0.412

Error Analysis:

```
{1: [{'Company': 'bp',
   'Pred value': array([ 0.03432425], dtype=float32),
   'Sentence': 'bp reports biggest ever annual loss',
   'True value': -0.98999999999999999,
   'index': 415},
  {'Company': 'glencore',
   'Pred value': array([ 0.03432425], dtype=float32),
   'Sentence': 'glencore shares in record crash as profit fears grow',
   'True value': -0.97099999999999997,
   'index': 142},
  {'Company': 'bp',
   'Pred value': array([ 0.03432425], dtype=float32),
   'Sentence': 'oil giant bp reports loss of $4.4 billion in 4th quarter of 2014',
   'True value': -0.96299999999999997,
   'index': 83},
  {'Company': 'aviva plc',
   'Pred value': array([ 0.03432425], dtype=float32),
   'Sentence': 'aviva posts forecast-beating 2015 operating profit of $3.8 bln',
   'True value': 0.94599999999999995,
```

BLSTM Sentence Representation:

1)Sentences are fixed length.

2)All words are represented as vectors.

Advantages of LSTMS:

a) Good at learning sequential data
b) Able to learn long term dependenci

Acknowledgement

This report is based on the works of Sudipta Kar and Andrew Moore.

https://github.com/apmoore1/semeval/blob/master/paper/lancaster-semeval-2017.pdf

http://www.aclweb.org/anthology/S17-2150