Sentiment Analysis of Company Earnings Conference Calls with Long Short-Term Memory

Shafiq K. Ebrahim [ebrahims@stanford.edu]

CS 230

Abstract

We examine the sentiment expressed during quarterly company earnings conference calls. Most sentiment analysis studies in the finance and accounting literature use techniques that ignore the order of words as well as the context of the information. Sequence modeling based on deep neural networks offer us the opportunity to capture the context in a more meaningful way. We compare the performance of the long short-term memory model (LSTM) with those of standard machine learning techniques. Our LSTM model struggles to outperform shallow machine learning models which suggests that a different deep learning architecture might be better suited to analyzing feature sets consisting of extremely long sequences of words.

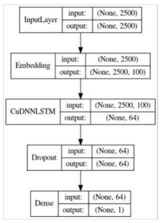
Data

- Quarterly earnings conference call transcripts from S&P Capital IQ for the period from January 2008 to December 2017.
- Focus only on U.S. companies belonging to the Russell 1000 Index.
- Extract all the components of text corresponding to questions by analysts and combine them together for each call. We do the same for all the responses by management.
- 36,174 examples. Average of 3,718 words each for answers and 1,466 words each for questions.

Abnormal Stock Returns

- Classification of the sentiment expressed by management or analysts is difficult because ground truth labels are not readily available.
- Manual labeling is not practical with a large data set and is prone to biases.
- Instead, we compute the conference call period return of each stock in excess of the Russell 1000 Index over a four-day window beginning on the date of the earnings conference call.
- We label each earnings conference call as positive or negative based on the sign of this abnormal return.

Models



- We use a long short-term memory model (LSTM) which is a type of recurrent neural network (RNN).
- Inputs are words from the conference call transcript. We use the pre-trained 100-dimension GloVe embeddings (400K vocabulary) without any additional fine-tuning (Pennington et al. (2014)).
- Single LSTM hidden layer with 64 units and a dropout rate of 0.8 followed by a dense layer with sigmoid activation function.
- We optimize the network (binary cross-entropy loss function) using the Adam algorithm with a learning rate of 0.0001 and a batch size of 32.
- Stop training after 30 epochs because of overfitting.

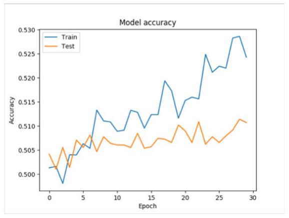
- We compare the results of our LSTM with three models – a random guess baseline based on the training set classification probability and two standard machine learning algorithms – naïve Bayes and support vector machines.

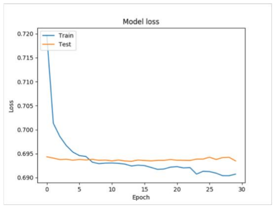
- We split our dataset into training, validation, and test sets (64%/16%/20%). We tune our LSTM based on results from the validation set.
- We tune the hyperparameters of the standard machine learning algorithms with 5-fold time-series cross-validation.

Results

Component	Answers		Questions	
	Training set	Test set	Training set	Test set
Baseline	0.500	0.497	0.496	0.496
Naïve Bayes	0.543	0.530	0.574	0.545
SVM	**0.577**	**0.547**	**0.612**	**0.571**
LSTM	0.531	0.495	0.503	0.501

Bold figures are highest values in each column.





Discussion and Future Work

- Training the LSTM model proved to be very challenging. The network appeared to be underfitting the data initially, so we added capacity by increasing the number of hidden layers from one to three and the number of hidden units up to 256. The model had difficulty achieving training and validation set accuracy above 50%.
- We hypothesized that the input sequences may have been too long for the network to process effectively (maximum length of over 18,000 words!), so we applied a cap of 2,500 words for each document.
- Then, we were experienced overfitting: the training set accuracy increased to about 70%, but the validation set accuracy fell back to 50% after initially increasing to about 52%.
- Using L2 regularization and trying the gated recurrent unit (GRU) architecture didn't help.
- Perhaps an alternative architecture such as a convolutional neural network (CNN) or the training of even lower dimensional word embeddings based on our own corpus might yield more promising results.

References

Pennington, J., Socher, R. & Manning, C. (2014) Glove: Global Vectors for Word Representation. *EMNLP* **14**, 1532-1543.