Economics Bulletin

Volume 40, Issue 2

Does Applying Deep Learning in Financial Sentiment Analysis Lead to Better Classification Performance?

Cuiyuan Wang
CUNY Graduate Center

Tao Wang
Queens College and CUNY Graduate Center

Changhe Yuan
Queens College and CUNY Graduate Center

Abstract

Using a unique data set from Seeking Alpha, we compare the deep learning approach with traditional machine learning approaches in classifying financial text. We apply the long short-term memory (LSTM) as the deep learning method and Naive Bayes, SVM, Logistic Regression, XGBoost as the traditional machine learning approaches. The results suggest that the LSTM model outperforms the conventional machine learning methods on all metrics. Based on the t-SNE graph, the success of the LSTM model is partially explained as the high-accuracy LSTM model distinguishes between positive and negative important sentiment words while those words are chosen based on SHAP values and also appear in the widely used financial word dictionary, the Loughran-McDonald Dictionary (2011).

We would like to thank the referee for excellent comments. All remaining errors are ours.
Citation: Cuiyuan Wang and Tao Wang and Changhe Yuan, (2020) "Does Applying Deep Learning in Financial Sentiment Analysis Lead to Better Classification Performance?", *Economics Bulletin*, Volume 40, Issue 2, pages 1091-1105
Contact: Cuiyuan Wang - cwang3@gradcenter.cuny.edu, Tao Wang - tao.wang@qc.cuny.edu, Changhe Yuan - changhe.yuan@qc.cuny.edu.
Submitted: November 17, 2019. **Published:** April 29, 2020.

1 Introduction

The ability to extract sentiment information from financial articles has always been appealing to researchers and market practitioners since market sentiment could predict asset prices ((Das and Chen, 2007), (Tetlock, 2007), (Chen et al., 2014)). In the past, researchers have been relying on dictionary-based sentiment analysis to measure the tone of a text. This method relies heavily on a pre-defined list (or dictionary) of sentiment-laden words. Nevertheless, pre-defined lists are either too generic to be suitable for financial texts, e.g., the Harvard IV-4 dictionary[1], or sample specific, e.g., the Loughran-McDonald Dictionary (hereafter LM Dictionary) (Loughran and McDonald, 2011) which is based on 10-k files. As a result, machine learning methods become popular in sentiment analysis as they could be used on classifying all kinds of financial texts.

Previous studies use various machine learning techniques to study sentiment from financial texts. Those related machine learning techniques include Naïve Bayes ((Antweiler and Frank, 2004), (Das and Chen, 2007), (Li, 2010), (Bartov et al., 2018), (Huang et al., 2014)), Support Vector Machine (Antweiler and Frank, 2004), and discriminant-based classifier (Das and Chen, 2007).

Recently, deep learning neural networks made a breakthrough and produced state-of-the-art results in many domains. As compared to "shallow" neural networks, deep learning is the application of neural networks to learning tasks using the complicated structures, which tends to capture the inherent dependencies among features. Due to the improvement of computing power and the availability of training data, the deep learning approach becomes possible in many fields. In particular, with the emergence of enormous user-generated content on social media and websites, it is now feasible to conduct computational studies on individual opinions or sentiments using deep learning approaches (Zhang et al., 2018).

In this paper, we study whether applying the deep learning approach in financial sentiment analysis leads to better classification performance as compared with traditional machine learning techniques. We apply both deep learning and traditional machine learning techniques on classifying stock opinion articles from the Seeking Alpha (hereafter, SA) website and compare their performance. SA is a popular investment social media platform with millions of registered users. Our selection of SA articles as the focus of this study is motivated by its popularity among investors and one unique feature of selected SA articles. The uniqueness of this selected group of SA articles is that those articles have already been categorized as bullish or bearish ideas by the SA website. In all previous studies using machine learning techniques, manual sentiment annotations of sentences or articles are necessary, which could lead to human judgment errors. In our case, we avoid this step and concentrate on the performance comparison among different machine learning methods.

Our results suggest that the deep learning model, Recurrent Neural Network (Rumelhart et al., 1986) with long short-term memory (Hochreiter and Schmidhuber, 1997), beats all the other machine learning methods including SVM, Naïve Bayes, Logistic Regression, and XGBoost in terms of classification accuracy, precision, recall, and area under curve (AUC). The Long Short Term Memory (LSTM) model achieves 95% classification accuracy, and the AUC is 0.97.

Furthermore, we find that the performance of the LSTM model is partially explainable via

[1]Please see http://www.wjh.harvard.edu/ inquirer/homecat.htm.

important features or words.[2] To visualize how the best LSTM model with the highest classification accuracy is related to important features extracted from articles, we plot important features on the t-SNE graph (van der Maaten and Hinton, 2008). The t-SNE graph from the LSTM model with high classification accuracy shows two separate clusters between positive and negative sentiment features. The LSTM models with low and medium accuracy do not separate positive and negative sentiment features as cleanly on their respective t-SNE plot.

Overall, the paper is among the first in examining whether applying the deep learning approach in financial sentiment analysis leads to better classification performance as compared with traditional machine learning techniques. The results suggest that applying the LSTM model to online financial documents improves classification accuracy. Moreover, the LSTM model with high classification accuracy could differentiate between positive features and negative features, which helps understand why the good LSTM model performs well in sentiment classification.

The rest of the paper is as follows. Section 2 introduces the data set from the SA website. Section 3 evaluates the performance of various classification methods. We also investigate the intersected important features between those extracted by the LSTM models using SHAP values and the modified LM dictionary, then analyze the word vectors of these features using t-SNE plots. Section 4 concludes.

2 Data

The paper studies articles downloaded from the Seeking Alpha website. SA is an online crowdsourced content service provider for financial markets. The website derives its content from independent contributors who could be stock analysts, traders, economists, academics, financial advisors, and industry experts. Articles submitted to SA are subject to editorial changes. The review process intends to improve the quality of submitted articles without interfering with authors' opinions, i.e., whether articles are bullish or bearish, are determined by contributing authors and not by SA editors.

SA contributing authors usually express their opinions on stocks. Some articles contain ideas about multiple stocks, while others only present single stock discussions. In this paper, we focus on single stock articles. SA allows authors not only to write articles but also to post comments in response to articles. In order to avoid the interference of comments posted after an article, we ignore all comments in our data set. SA categorizes some of the articles to be bullish-idea and bearish-idea articles. In the paper, we only study this unique group of articles. Given the information, we should be able to identify the sentiment of those articles. Those articles labeled as bullish ideas are classified as bullish articles and have positive sentiment. Those articles tagged as bearish ideas are classified as bearish articles and have negative sentiment.

We download all opinion articles published from January 1, 2006 to December 31, 2015 on the SA website. After removing the articles on multiple stocks, ETFs, and ETNs, we have 157818 single-ticker articles in total. Among those downloaded single-stock articles, 60418 articles are marked as bullish or bearish ideas, and the rest 97400 articles do not have labels. As discussed earlier, with this unique group of labeled articles, we can directly evaluate the classification per-

[2] In this paper, we use "features" and "words" interchangeably.

formance among traditional machine learning methods and deep learning methods. Among those labeled single-stock articles, 52461 articles are bullish ideas, and 7957 articles are bearish ideas. The ratio of the number of bullish articles to bearish articles is 6.59 to 1, suggesting that the majority of articles in our sample show positive sentiment. Table 1 presents the summary statistics of the articles used in the study. The mean number of words in each article is 1187, and the median number is 982, which is the input length for the embedding layer of the LSTM model.

3 Experiment and Results

3.1 Setup

We apply the Support Vector Machine (Cortes and Vapnik, 1995), Naïve Bayes (Friedman et al., 1997), Logistic Regression (Peng et al., 2002), and XGBoost (Chen and Guestrin, 2016) packages from sklearn in Python and set the class weight to be the ratio (6.59:1) between positive and negative classes. We adopt the linear kernel for the SVM model and implement the MultinomialNB as our Naïve Bayes model. The XGBoost model trains for one hundred rounds in total until the validation loss shows no improvement for ten rounds. In the experiments, we use 80% of the labeled articles as the training data and the rest as testing data for all the models.

For the deep learning model, we choose LSTM, a variation of the RNN model. The LSTM model is a sequential model that could capture long-term dependency from the context, which is the natural choice for textual analysis. In the experiment, we set the class weight to be the ratio of positive and negative classes.

We conduct a simple sequential LSTM model with three layers. The first layer is the embedding layer that encodes the word vectors of all features. We use the Word2Vec embeddings (Mikolov et al., 2013) in the input layer of LSTM, representing words in high dimensional numeric vector forms. Initially, we assign the vector with random numbers. After training through the neural network, the learned vectors aggregate similar semantic words in the vector space. By using the Word2Vec embedding method, we capture the inherent notion of similarity. Gentzkow et al. (2019) has detailed discussions on embeddings techniques, including the Word2Vec embedding method.

We set the embedding size to be 32 dimensions, i.e., convert each word into a 32-dimension word vector. The input length is set to be the median length (982 words) of all articles. Any articles with less than 982 words are padded with 0 at the end. Any articles with more than 982 words are truncated at the first 982 words. In the second layer, we adopt the Long Short Term Memory mechanism. The number of hidden units is 128, and the drop out rate is 0.3 to avoid overfitting. In the last layer, we apply the softmax activation function to normalize the output to be probabilities of each class.

We monitor the accuracy and validation loss of the model using the training and validation data in each epoch. The model is set to run for 40 epochs, and the batch size is 128 in each epoch. In order to attenuate the issue of overfitting, we end the model when the validation loss stops declining for ten epochs. The best overall model is obtained at epoch 33.

The subsequent section analyses the results from the traditional machine learning methods and the LSTM model.

3.2 Results

Table 2 presents the results from the Support Vector Machine, Naïve Bayes, Logistic Regression, XGBoost, and LSTM on the SA data set. The results reveal the strong performance of the LSTM model compared to other models. In terms of overall performance, accuracy for the best LSTM model generated at epoch 33 is 0.95, which means among all articles, 95 percent of the articles are classified correctly. In comparison, accuracies for the SVM, Naïve Bayes, Logistic Regression, and XGBoost are lower and at 0.91, 0.86, 0.92, 0.86 respectively. The weighted F1, a weighted measure of accuracy, for LSTM is 0.95 while they are 0.91, 0.87, 0.92, and 0.88 for the SVM, Naïve Bayes, Logistic Regression, and XGBoost respectively. The Area Under Curve (AUC), another performance measure and illustrates how much a model is capable of distinguishing between classes, is 0.97 for the LSTM model and 0.92, 0.84, 0.93, and 0.94 for the SVM, Naïve Bayes, Logistic Regression, and XGBoost models respectively. Figure 1 plots the ROC-AUC graph. Obviously, the LSTM model outperforms all other models dramatically. Overall, the LSTM model performs the best, not only in overall accuracy but also in weighted F1 and Area Under Curve, which are performance criteria for machine learning methods.

Concerning specific criterion for each class, for example, the positive recall, the percentage of correctly identified bullish articles among all bullish articles, is the highest for the LSTM model at 0.96 while at 0.95, 0.89, 0.95, and 0.86 for the SVM, Naïve Bayes, Logistic Regression, and XGBoost models respectively. Negative recall, the percentage of correctly identified bearish articles among all bearish articles, is 0.87 for the LSTM model while they are 0.68, 0.65, 0.72, and 0.86 for the SVM, Naïve Bayes, Logistic Regression, and XGBoost models respectively. These numbers suggest that recall positives are all higher than recall negatives among all models except that they are the same for the XGBoost model. The result is not surprising given that the ratio of the number of bullish articles to bearish articles is 6.59 to 1, and most machine learning methods could over-classify majority class, especially when the data set is very unbalanced. The only exception here is the XGBoost model, which is capable of balancing the recall between positive and negative classes. Still, by balancing between positive and negative classes, its overall accuracy suffers at 0.86, lower than all other models except the same as that from the Naïve Bayes model.

We also present the runtime for each machine learning method, the runtime for the SVM, Naïve Bayes, Logistic Regression, and XGBoost models are 40137, 36, 91, and 105 seconds respectively.[3] The Naïve Bayes method, the most popular used machine learning method so far in accounting and finance,[4] has the fastest runtime, but its accuracy, weighted F1, and AUC are all among the lowest. The SVM model is computationally expensive due to the core of the SVM is a quadratic programming problem which scales with the number of samples. Compared with the computational speed of the SVM model, the LSTM model runs a little bit faster at 33,900 seconds.

The overall picture from Table 2 is that the LSTM model performs the best among all other models on almost every metric. The reasons why the LSTM model performs the best are that the LSTM model captures sequential correlations among words in recurrent mechanism and also encodes more context information than the traditional machine learning approaches based on raw counts of all the features. The count vectorizer used by traditional machine learning methods con-

[3] All the codes are run with Python sklearn under the 2017 MacBook Pro with 3.5 GHz Intel Core i7 7567U and 16 GB RAM.

[4] Please see Loughran and McDonald (2016).

tains only the count of the features, which ignores correlations among features. On the other hand, many deep learning models such as LSTM, update learned word embeddings as input features in every learning step.

In order to examine whether the performance of the LSTM model is partially explainable via important features or words, we extract important features from LSTM models using SHAP values and tag the polarity of the features using positive and negative word lists from the financial dictionary, the LM Dictionary(Loughran and McDonald, 2011). As the LM dictionary was created based on 10-k filings, our data set is from online financial opinion text, we add three words ("long", "short", "undervalued") to the dictionary, which appear as important features from the best LSTM model. Once important features are extracted from LSTM models, we can visualize how related these features are using the t-SNE graph.(van der Maaten and Hinton, 2008)

SHAP (SHaply Additive exPlanations) (Lundberg and Lee, 2017) is a unified approach to interpret model predictions.[5] SHAP assigns each feature an importance score for each observation (e.g. article), we accumulate the SHAP values grouped by feature names to obtain a global importance value for each feature. In our experiment, we extracted the topmost 300 features to intersect with the modified word list from the LM Dictionary to identify their polarity.

Table 3 lists all the intersected words ordered by SHAP values for the best LSTM model.[6] For the best LSTM model, the number of common features is 41. Among them, there are 28 positive features and 13 negative features. Note that LSTM models with low and medium classification accuracy generate fewer common features with the modified LM dictionary.

The LSTM model keeps adjusting the weights of the embedding layer as it learns. Subsequently, from the embedding layer of the LSTM model, we extract word embeddings for these features based on the best LSTM model. To visualize how the best LSTM model is related to important features extracted from the articles, we transform the learned word embeddings of these important features and plot them on the t-SNE graph. The t-Distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear dimensionality reduction technique used to simplify large data sets graphically. The t-SNE technique preserves the structure of the data points in the high-dimension space and converts them to be presentable on the two-dimension space. In t-SNE, close data points in the high dimensional space lump together in the low-dimensional space as well, and the unrelated data points will be far apart when projected into the low-dimensional space. Many deep learning models, such as the LSTM model, need learned word embedding results as input features. Word embedding techniques transform words in vocabulary to vectors of continuous real numbers, or loosely speaking, word embeddings are vector representations of a particular word. Since vector representation of each word is of many dimensions, t-SNE could be used to reduce those dimensions and plot them on the t-SNE graph.

For the t-SNE graph, as we only choose these important features that come with positive or negative polarity based on the modified LM Dictionary, we can check how these features are related. In Figure 2, the LSTM model with high classification accuracy (accuracy = 0.95) shows two separate clusters where almost all positive sentiment features cluster together on the left, whereas all negative sentiment features cluster on the upper right corner. Thus, the best LSTM model is capable of separating positive features from negative features. In contrast, the LSTM model shown

[5]We thank the referee for the suggestion on using SHAP to identify important features.

[6]The best LSTM model has not only the highest accuracy but also the highest weighted F1 and AUC.

in Figure 3 with medium accuracy (accuracy = 0.76) could not separate positive and negative sentiment features as cleanly as the LSTM model with high classification accuracy. The low-accuracy (accuracy=0.52) LSTM model from Figure 4 also could not show any clusters clearly either. Furthermore, LSTM models with low and medium classification accuracy generate far fewer common features with the modified LM dictionary.

Figures 2 to 4 suggest that separation from the clusters of the word embeddings becomes evident as the classification accuracy of LSTM models improves. Thus, we can conclude that the best LSTM model with high classification accuracy has a better representation of the word embeddings, which is consistent with the idea that the best LSTM model is partially explainable from its identification on important features.[7]

4 Conclusion

This study evaluates the performance of different machine learning models on a finance social media data set. The results show that the deep learning method, Long Short Term Memory Model (LSTM), outperforms all the other traditional machine learning methods, including Naïve Bayes, SVM, Logistic Regression, and XGBoost on almost all metrics. To understand and verify the classification ability of the LSTM model, we extract important features generated by SHAP and tagged those features using the financial word list, the modified LM dictionary. Among those important features with polarity, we found that similar features clustered together, and dissimilar features were far apart after projecting the word embeddings of the best LSTM model using t-SNE, which partially explains why the best LSTM model has the highest classification accuracy. Our paper suggests that the LSTM model could perform well in identifying financial sentiment with explainable features.

[7]We also examine the pre-trained word vectors from the Gensim implementation of Word2Vec (Mikolov et al., 2013) trained on our dataset. The result suggests that the Word2Vec algorithm alone is not able to separate positive features from negative features on the t-SNE plot. The result is available upon request.

Bibliography

Antweiler, W. and Frank, M. Z. (2004). Is all that talk just noise? the information content of internet stock message boards. *The Journal of Finance*, 59(3):1259–1294.

Bartov, E., Faurel, L., and Mohanram, P. (2018). Can twitter help predict firm-level earnings and stock returns? *The Accounting Review*, 93(3):25–57.

Chen, H., De, P., Hu, Y. J., and Hwang, B.-H. (2014). Wisdom of crowds: The value of stock opinions transmitted through social media. *The Review of Financial Studies*, 27(5):1367–1403.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Das, S. R. and Chen, M. Y. (2007). Yahoo! for amazon: Sentiment extraction from small talk on the web. *Management Science*, 53(9):1375–1388.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29:131–163.

Gentzkow, M., Kelly, B., and Taddy, M. (2019). Text as data. *Journal of Economic Literature*, 57(3):535–74.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Huang, A. H., Zang, A. Y., and Zheng, R. (2014). Evidence on the information content of text in analyst reports. *The Accounting Review*, 89(6):2151–2180.

Li, F. (2010). The information content of forward-looking statements in corporate filings—a naïve bayesian machine learning approach. *Journal of Accounting Research*, 48(5):1049–1102.

Loughran, T. and McDonald, B. (2011). When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *The Journal of Finance*, 66(1):35–65.

Loughran, T. and McDonald, B. (2016). Textual analysis in accounting and finance: A survey. *Journal of Accounting Research*, 54(4):1187–1230.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

Mikolov, T., Corrado, G., Chen, K., and Dean, J. (2013). Efficient estimation of word representations in vector space. pages 1–12.

Peng, C.-Y. J., Lee, K. L., and Ingersoll, G. M. (2002). An introduction to logistic regression analysis and reporting. *The Journal of Educational Research*, 96(1):3–14.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Tetlock, P. C. (2007). Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168.

van der Maaten, L. and Hinton, G. (2008). Viualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Zhang, L. J., Wang, S., and Liu, B. (2018). Deep learning for sentiment analysis: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 8.

Table 1: Summary Statistics: Seeking Alpha Single Ticker Articles

Year	Bullish (Positive)	Bearish (Negative)	Ratio	Mean No. of Words Per Article	Median Length of Article	Standard Deviation
2006-2015	52461	7957	6.59 : 1	1187	982	867

- Articles were published from January 1, 2006 to December 31, 2015 at http://www.seekingalpha.com.
- Articles studied were labelled either as bullish ideas or bearish ideas by SA.
- Articles with bullish ideas are 87% of the sample while articles with bearish ideas are 13% of the sample.

Table 2: Classification on SA Articles: Evaluation of Machine Learning Approaches

Method	TP	FP	TN	FN	Precision Positive ($\frac{TP}{TP+FP}$)	Precision Negative ($\frac{TN}{TN+FN}$)	Recall Positive ($\frac{TP}{TP+FN}$)	Recall Negative ($\frac{TN}{TN+FP}$)	Accuracy	Weighted F1	AUC	RunTime(Seconds)
Panel A: Raw Count Vectorizer												
SVM	10019 (95%)	486 (32%)	1024 (68%)	555 (5%)	0.95	0.65	0.95	0.68	0.91	0.91	0.92	40,137
NB	9394 (89%)	522 (35%)	988 (65%)	1180 (11%)	0.95	0.46	0.89	0.65	0.86	0.87	0.84	36
LR	10021 (95%)	416 (28%)	1094 (72%)	553 (5%)	0.96	0.66	0.95	0.72	0.92	0.92	0.93	91
XGBoost	9122 (86%)	207 (14%)	1303 (86%)	1452 (14%)	0.98	0.47	0.86	0.86	0.86	0.88	0.94	105
Panel B: LSTM model												
LSTM	9984 (96%)	213 (13%)	1452 (87%)	435 (4%)	0.98	0.77	0.96	0.87	0.95	0.95	0.97	33,900

- SVM, NB, LR, and XGBoost stand for Support Vector Machine, Naïve Bayes, Logistic Regression, and eXtreme Gradient Boosting methods respectively. TP, FP, TN, FN stand for true positive, false positive, true negative, and false negative respectively.
- LSTM represents Long Short Term Memory model. The best model was obtained at epoch 33.
- 80% data are randomly chosen as the training data, the rest as testing data.
- Numbers in parentheses correspond to sensitivity, fall-out, specificity, and miss rate. sensitivity=$TP/P = TP/(TP+FN)$, fall-out=$FP/N = FP/(FP+TN)$, specificity=$TN/N = TN/(TN+FP)$, miss rate=$FN/P = FN/(FN+TP)$.
- Accuracy is defined as $\frac{TP+TN}{TP+FP+TN+FN}$.
- $F1_{weighted} = 2 * \frac{Precision_{weighted} * Recall_{weighted}}{Precision_{weighted} + Recall_{weighted}}$, where $Precision_{weighted} = \frac{n_1 * Precision_{Pos} + n_2 * Precision_{Neg}}{n_1 + n_2}$, $Recall_{weighted} = \frac{n_1 * Recall_{Pos} + n_2 * Recall_{Neg}}{n_1 + n_2}$, and n_1 and n_2 are the total numbers of bullish and bearish articles respectively.
- AUC is the area under the ROC curve.

Table 3: Features Ranked by SHAP Values from the best LSTM Model (Accuracy = 0.95)

Model	Positive features	Negative features
LSTM	long* undervalued* strong opportunity attractive improve positive good excellent benefit reward able impressive winner advantage happy great outperform improved boost profitable advantages stable opportunities strength confident valuable attain	short* overvalued decline negative opportunistic weak declining questionable worse unable failed unfortunately disagree

- All words are ranked in the order of importance obtained by SHAP values.
- Three words "long", "short", "undervalued" based on SHAP values are added but do not appear in the LM dictionary (Loughran and McDonald, 2011).
- All other words are those important words from the best LSTM model and also appear in the LM dictionary.
- Fewer features are obtained from LSTM models with low and medium accuracy. Please see Figures 3 and 4.

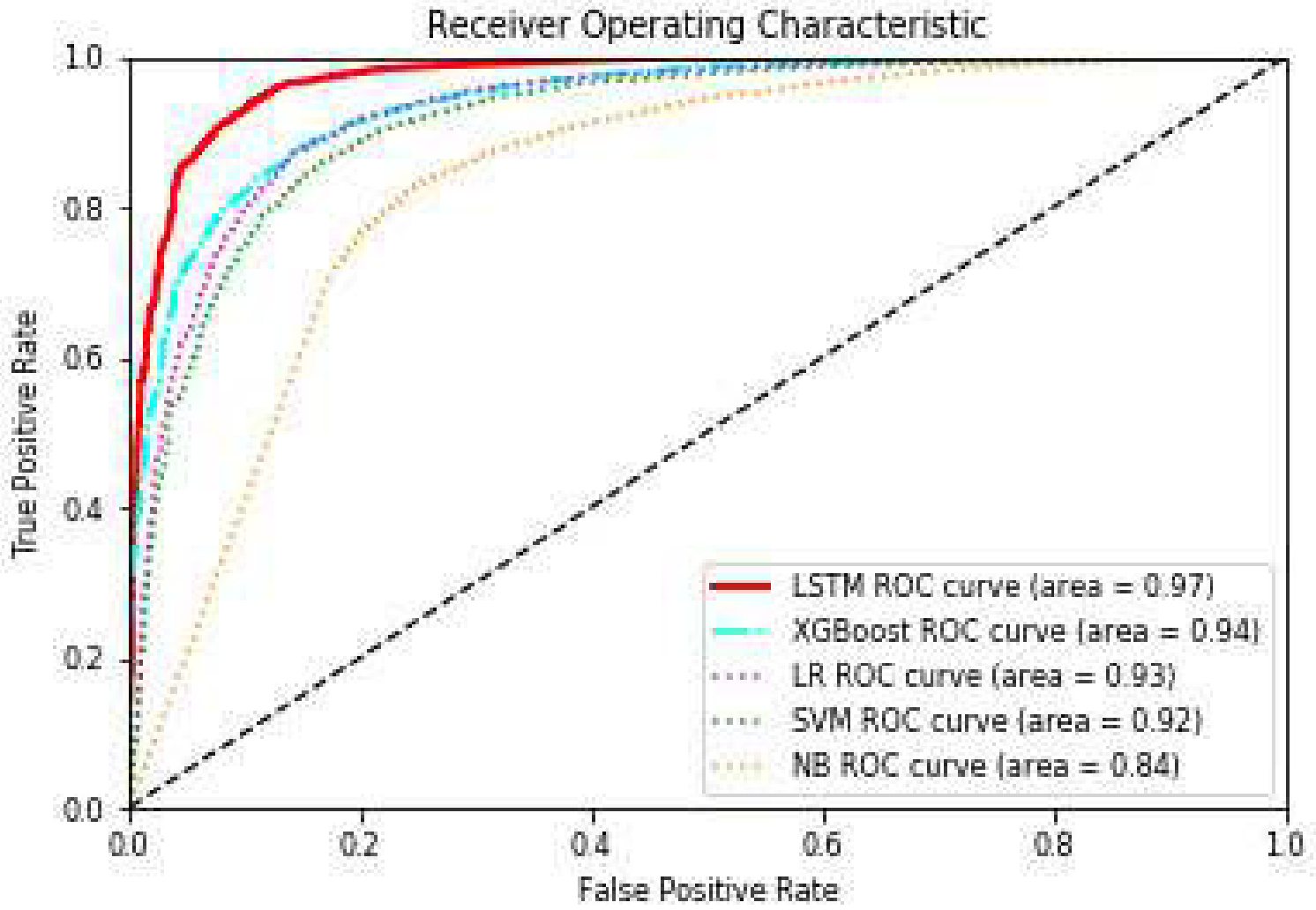

Figure 1: A comparison of the ROC AUC scores among different machine learning methods on the Seeking Alpha data set

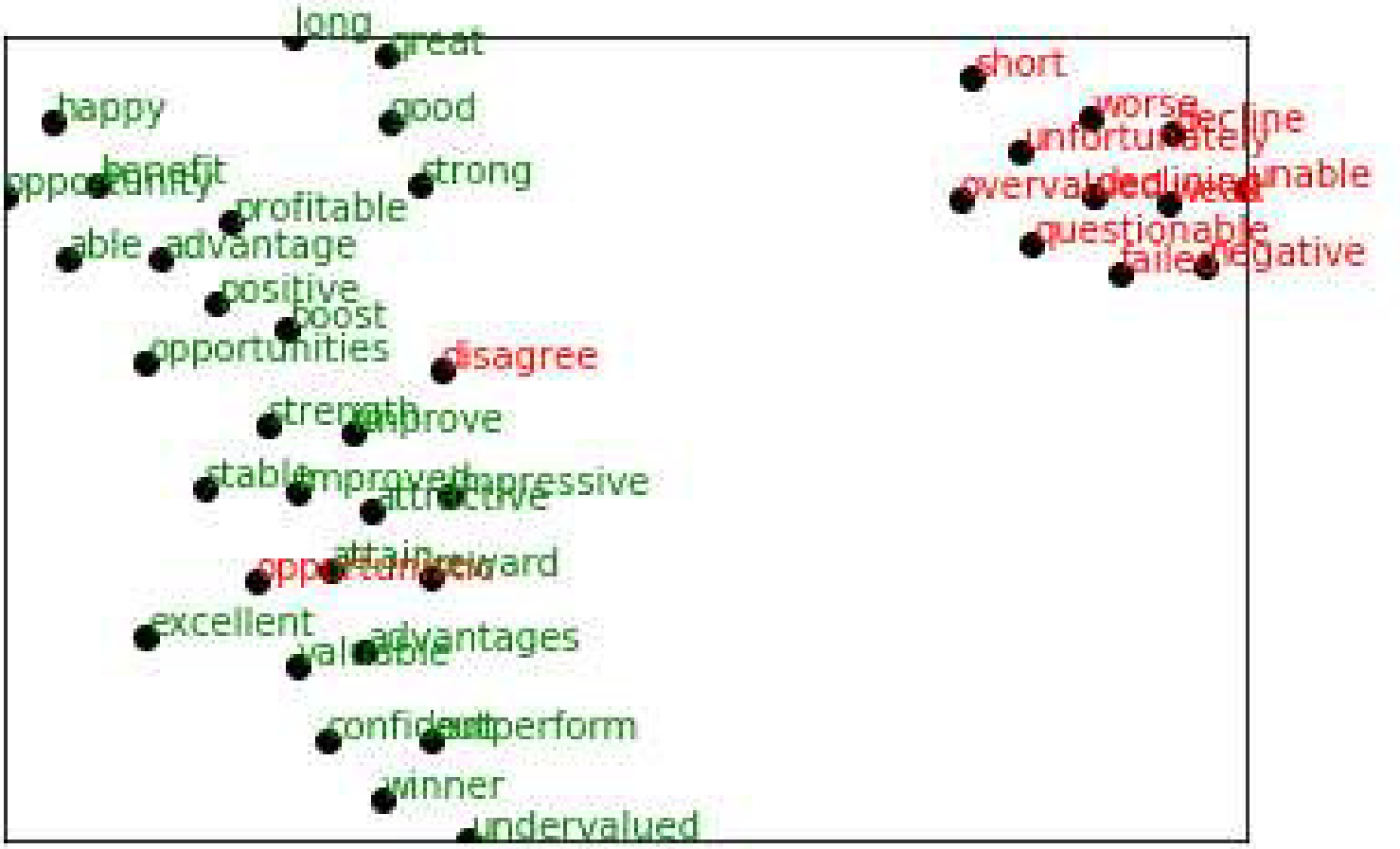

Figure 2: t-SNE Plot for Sentiment Words from the LSTM model (Accuracy = 0.95)
(Sentiment words are important words based on SHAP values from the model and also appear in the modified LM dictionary.)

Figure 3: t-SNE Plot for Sentiment Words from the LSTM model (Accuracy = 0.76)

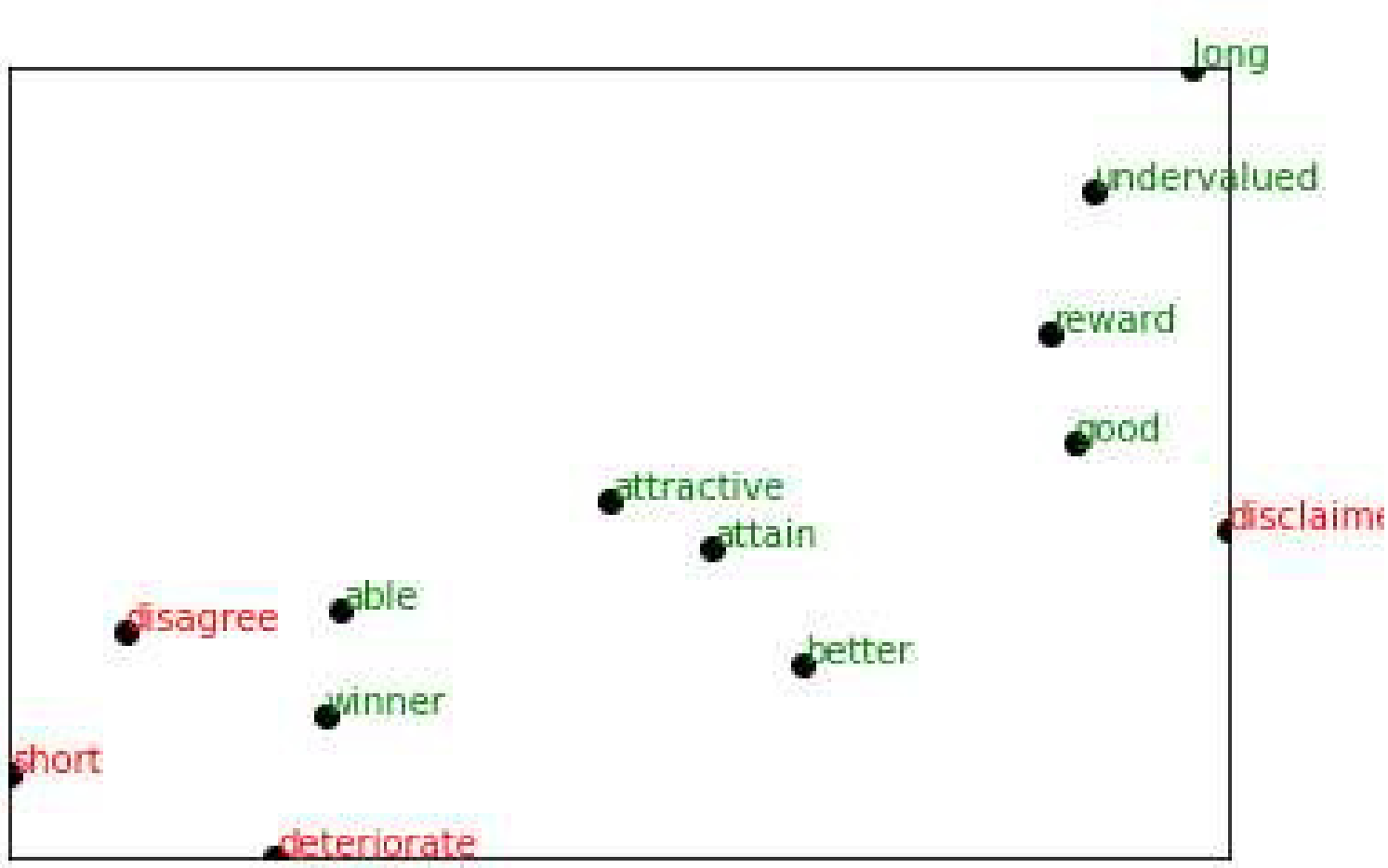

Figure 4: t-SNE Plot for Sentiment Words from the LSTM model (Accuracy = 0.52)