Sentiment Analysis Using Support Vector Machines and Long Short Term Memory Neural Networks - A Critical Evaluation

A. Ritchie, D. Sikar - INM427 Coursework - PT1 MSc Data Science *

March 2019

Abstract

Sentiment analysis is one of the most active research areas in natural language processing. Being able to quickly and accurately identify sentiment has proven to be beneficial for businesses and organizations across numerous sectors. This paper provides a critical evaluation of two neural computing models on performing sentiment analysis of consumer review text. The models considered are Support Vector Machines (SVMs) and Long Short-Term Memory (LSTM), which are fine-tuned utilizing a grid-search function on the hyper-parameters specific to each model.

Keywords— Sentiment Analysis, Support Vector Machines, Long Short-Term Memory, Deep Learning

1 Introduction

Sentiment analysis is one of the most active research areas in natural language processing and machine learning research [8][11][14]. With the growth of social media, review sites, and other online resources, there are vast amounts of opinions, comments, recommendations, and reviews shared every day. Being able to quickly and accurately determine sentiment of this data is beneficial to organizations and businesses alike. This allows organizations to target their audience, make intelligent recommendations, and identify new market opportunities [2].

Initial attempts to classify text used dictionaries and bag of words representations of documents. These methods, however, tend to lose the context and meaning often associated with sentences. This can be especially difficult to handle with polarized sentences. Recent studies have focused on creating vector representations for words and documents that are able to handle sparse and discrete features in text classification. Even more recently, deep learning models have been used due to their ability to retain memory between training examples. This has allowed for more critical evaluation of text and the relationship of words in context.

This paper investigates the effectiveness of two neural network models, Support Vector Machines (SVM) and Long Short-Term Memory (LSTM), to determine sentiment of reviews from Amazon, Yelp, and IMDb. The hyperparameters of each model are fine-tuned utilizing a grid-search function to find the best fit models. The paper then goes on to critically evaluate findings, compare results and propose areas of further investigation.

1.1 Support Vector Machines (SVM)

SVMs have been used stochastically to compare classification error rate and likelihood scores [10]. Kernels may be used to add dimensionality and consequent separability to data which is the ultimate goal our a classifier. SVMs have been used standalone but increasingly in hybrid environments[5].

In this study an existing word vector space [6] is used to examine various dimensions of differences between word vectors. Pre-trained representations have been used as building blocks of many Natural Language Processing and Machine Learning applications, such as word2vec in NLP pipelines to improve their performance [7].

1.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is an adaptation of Recurrent Neural Networks (RNNs). Like RNNs, LSTM utilizes internal memory to process temporal dependencies in sequential data. RNNs, however, are limited in

*ashley.ritchie, daniel.sikar @city.ac.uk - City University of London

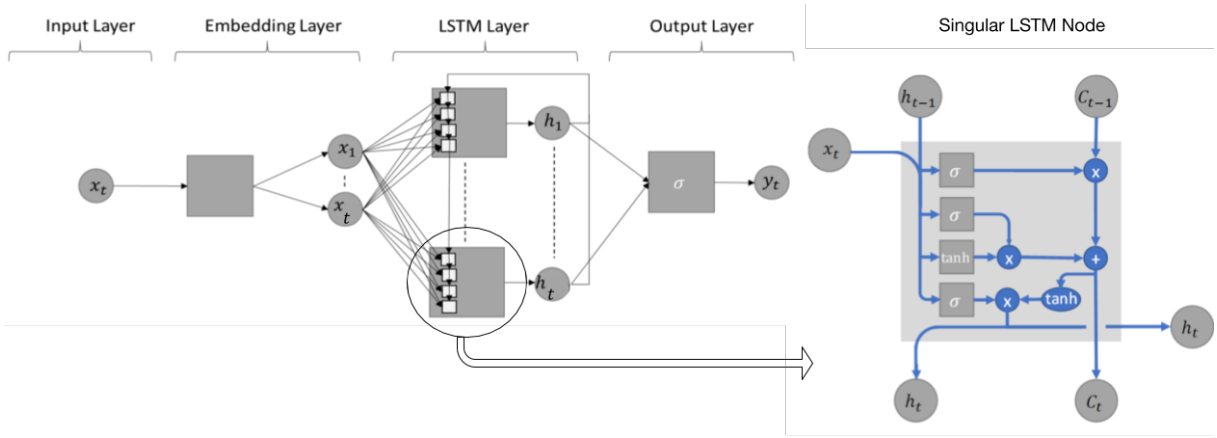

Figure 1: Long Short-Term Memory (LSTM) Network

such a way that the current output is determined only by the networks previous output. This creates a vanishing gradient problem. LSTM seeks to fix this by introducing activation states in the hidden layer.

LSTMs have the ability to add or remove information through these activation states, or gates. First the model reads input from the previous hidden state. It then determines what values to remove from memory through the forget gate. It also has an input gate, which controls the activations that will pass into the memory and an output gate that determines which activations will be passed back to the RNN. As such, LSTMs are able to extract important features from multiple previous time steps, giving it a longer memory. This makes LSTM appealing for sentiment analysis because the model is better able to capture long-term dependencies.

Adding gates into the hidden layer increases training time and requires more memory from the system. Additionally, this method requires significant parameter fine-tuning, is subject to overfitting, and brings into question whether certain parameters are even needed. Despite these pitfalls, LSTM has proven successful in outperforming many other commonly used learning algorithms when it comes to sentiment analysis[9][12][13].

1.3 Hypothesis

Due to the ability of LSTMs to capture long-term dependencies we believe this model will outperform SVMs in correctly identifying sentiment.

2 Dataset

Three distinct bodies of data were used in our analysis. The first consisting of consumer reviews which are common to LSTM and SVM approaches. The remaining two were used only for SVM analysis; positive and negative words and a pre-trained word embedding and sentence classification model.

2.1 Sentiment Labelled Sentences

The Sentiment Labelled Sentences Dataset [4] used in our evaluation was obtained from the UCI Machine Learning Repository. It consists of 3000 highly polar reviews labelled with positive or negative sentiment. The dataset was collected from three websites - Amazon, IMDb, and Yelp - and there exists 500 positive and 500 negative reviews from each website. Reviews are string data types, labels are discrete values of 0 and 1, representing negative and positive sentiments as categorical data.

From an initial view of the data as seen in Table 1, there appears to be an average of 15.04 tokens with a standard deviation of 41.33. Breaking it down by website, Amazon and Yelp appear to have similar structures, while IMDd reviews are longer on average and have a larger variation in length.

2.2 Positive and Negative Words

The positive and negative words dataset is based on prior work [3] where feature-based summaries of customer reviews of products are examined, then features are identified with respect to products that customers have expressed their opinions on features, finally lists of positive and negative sentiment words were created, consisting of string data types for sentiment words and ”Positive” and ”Negative” labels which are essentially categorial data.

Suggested rework of above sentence: The positive and negative words dataset is based on prior work [3] where feature-based summaries of customer reviews for products were examined. Features of the products

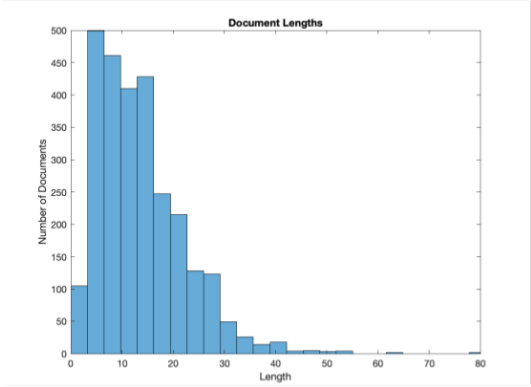

Figure 2: Document Token Distribution

Token Density

	Mean	Std. Dev.
All	15.04	41.33
Amazon	11.95	7.27
IMDb	22.41	77.94
Yelp	12.61	6.76

Table 1: Document Token Statistics

reviewed were identified and lists of positive and negative sentiment words were created. These lists consist of string data types for sentiment words and ”Positive” and ”Negative” labels which are essentially categorial data.

2.3 Word Embeddings

Word embeddings are vector representations of words. The embedding used in this study is the fastTextWordEmbedding matlab implementation, originally developed by Mikolov et al [7]. It adds several modifications to the standard word2vec training pipeline and significantly improves the quality of the resulting word vectors. The pre-trained word embedding for English words used is a 300-dimensional x 1 million (approximately) vector of *single* data types.

3 Methodology

For implementation and model comparison, the original dataset is divided using a 90/10 holdout method. Our training data (90%) is then used for training, validating, and fine-tuning hyperparameters on both the SVM and LSTM models. A grid-search methodology was employed for both models to identify optimal hyperparameters.

When working with text data, the first step in implementation is preprocessing the text. Initially preprocessing consisted of transforming text to lowercase, removing punctuation, stopwords, and short words, then normalizing the data. However through experimentation, both models performed better when limiting preprocessing to lowercasing text and removing punctuation. In regards to LSTM, this better performance can be attributed to the models ability to assess context, capture important parts of text and remove unimportant features.

The starting data set is the same for both methods. The processes then diverge as explained:

3.1 Support Vector Machine Methodology

To model a text sentiment analysis SVM, two additional datasets are required as described in 2.2 and 2.3.

First, both word sentiment datasets are compared against the pre-trained word embedding. Any sentiment words that do not exist in the word embedding vocabulary are excluded, as no meaningful vectors will be obtained by such words. The sentiment words are then vectorized and partitioned into training and testing sets, using a 10% holdout. An SVM model is then created based on the optimal parameters obtained via a grid search. Once the model has been obtained, customer reviews are then prepared by removing non-existing vocabulary words (w.r.t. pre-trained word embedding), as well as per procedure described in 3.

Finally, predictions are made by the model for every review. Each review is vectorized, resulting in a score which can then be compared against the original label in order to generate the accuracy for the chosen hyperparameters. Note, this final review-scoring step does not contain training and testing partitions. Scoring is performed on the entire review corpus. Scores greater than zero are considered positive, scores less or equal to zero are considered negative.

3.2 LSTM

The LSTM model in this paper utilizes an Adam optimisation on the training data. This solver uses gradient descent with adaptive learning rates on each parameter with momentum. This method improves performance on problems with sparse gradients such natural language processing[1]. A document embedding matrix was fed into our LSTM layer, which was then fed into the final softmax layer for sentiment classification.

During the implementation and fine-tuning of the LSTM model, many hyperparameters were considered. The encodings were set to be padded or truncated at 60 tokens, as most documents fell within this range. A grid-search was implemented for the hyperparameters of embedding dimensions, hidden layer dimensions, training epochs, and learning rate. The grid search then provides our model with the optimal parameters.

4 Results and Evaluation

SVM and LSTM use different hyperparameters that dictate best model and algorithm evaluation as seen in the hyperparameter grid search.

LSTM						SVM				
Embedding Dimension	Hidden Size	Max Epochs	Learning Rate	Accuracy	Time	Kernel Function	Polynomial Order	Box Constraint	Accuracy	Time
50	50	10	0.003	0.86131	14.182201	linear	N/A	1	0.7654	66
50	50	8	0.003	0.85036	11.184011	linear	N/A	0.01	0.7573	33
100	25	10	0.003	0.84672	11.741333	linear	N/A	0.02	0.7774	31
100	50	5	0.003	0.84672	8.61702	linear	N/A	0.03	0.7784	26
30	25	5	0.003	0.84307	5.105751	linear	N/A	0.04	0.7774	25
30	25	8	0.003	0.84307	6.856266	rbf	N/A	0.01	0.5	44
50	50	5	0.003	0.84307	6.654861	rbf	N/A	0.02	0.5	45
50	50	5	0.005	0.84307	6.960595	rbf	N/A	0.03	0.5	46
30	50	5	0.003	0.83942	5.976815	polynomial	2	N/A	0.769	25
50	50	10	0.005	0.83942	14.238041	polynomial	3	N/A	0.767	28
100	25	5	0.003	0.83577	6.087601	polynomial	4	N/A	0.7714	25
50	25	5	0.01	0.83212	5.120414	polynomial	5	N/A	0.775	25

Figure 3: Grid search

4.1 SVM Results and Evaluation

SVMs produced best results (highest accuracy) with linear kernel and a 0.04 box constraint parameters. While rbf kernels presented poor performance clamped at 50% accuracy, which may be explained by, ultimately, the binary classification problem addressed in this study. Polynomial kernels proved very effective, producing closer to highest accuracies, as well as producing the fastest results so that is a factor that may be considered for future reference, specially for online models.

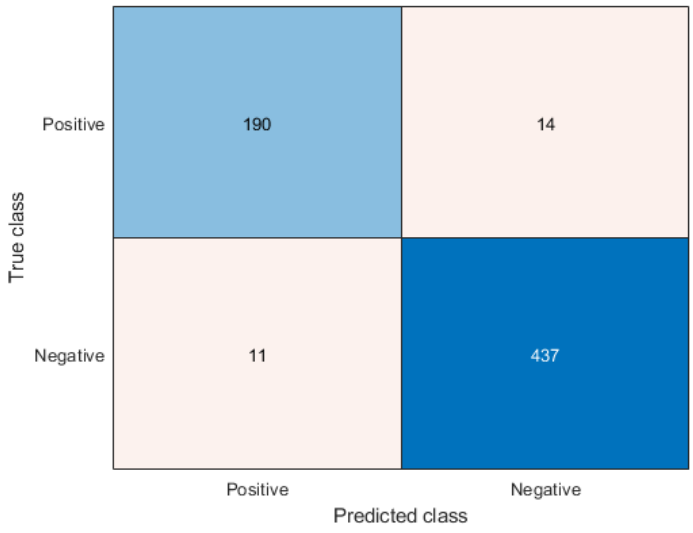

Figure 4: SVM best (linear kernel) model confusion matrix

4.2 LSTM Results and Evaluation

In training the dataset on the sentiment classification task, the optimal hyper-parameters were found to be 50 embedding dimensions, 50 hidden layer dimensions, 10 epochs, and a learning rate of .003. This yielded a testing accuracy of 86.13% During the fine-tuning and grid-search phase, it was found that increasing embedding dimensions and the hidden layer dimensions greatly increased calculation time while yielding similar accuracy to

that of lower dimensions. Similarly, it was found that increasing the number of epochs above 10 yielded greatly increased calculation times but did not improve accuracy. The learning rate was tested between .001 and .01 and was found to improve accuracy of the training data when set to .003.

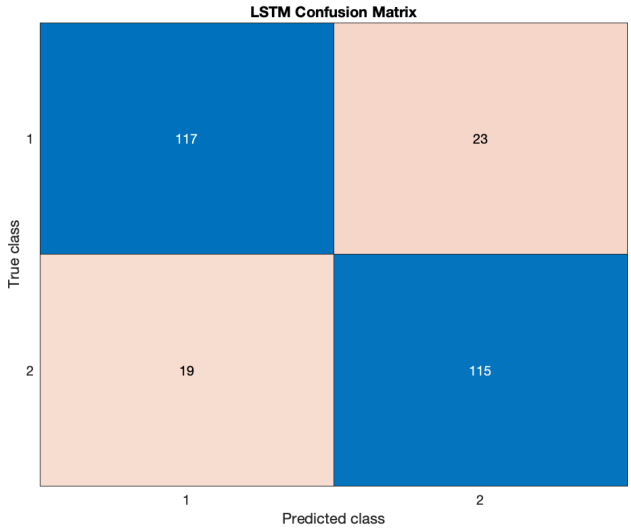

Figure 5: LSTM Confusion Matrix

Training the LSTM model proved to be somewhat complex as accuracies differed each time model was re-run, despite holding datasets and hyperparameters equal. This made fine-tuning parameters difficult as often times optimal parameters would change. For example, one iteration resulted in optimal parameters of 100 embedding dimensions, 15 hidden layer dimensions, 5 epochs (early stopping), and a learning rate of .03. Additionally training was time consuming as there were numerous hyperparameters that often varied greatly. Fitting the best model was a result of trial and error of training sets.

5 Conclusion

In this study we demonstrated how, with the use of word embeddings and a reference set of positive and negative keywords, texts can be classified by sentiment with SVM classifiers and LTSM networks. While this proves to be a successful method in sentiment classification, being able to look at context and long-term dependencies proves to outperform. LSTM is able to capture the context largely due to its ability to propagate important features and remove unimportant features over the longer training distances.

The best SVM model produced 77.84% accuracy while the best LTSM model produced 86.13% accuracy, which confirms our hypothesis that LSTMs outperform SVMs in identifying text sentiment.

With reference to the SVM classifier, future work would benefit from adding positive and negative sentiment words from the corpus to be analysed i.e. eliminating the intersect, excluding non-vocabulary words and adding remaining words to positive and negative sentiment word lists.

Acknowledgements

The authors wish to thank Abi Sowri for her guidance and support.

References

[1] J. Brownlee. Gentle introduction to the adam optimization algorithm for deep learning. July 2017.

[2] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 513–520. Omnipress, USA, 2011.

[3] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pp. 168–177, 2004. doi: 10.1145/1014052.1014073

[4] D. Kotzias, M. Denil, N. de Freitas, and P. Smyth. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pp. 597–606. ACM, New York, NY, USA, 2015. doi: 10.1145/2783258.2783380

[5] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pp. 142–150. Association for Computational Linguistics, Stroudsburg, PA, USA, 2011.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[7] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[8] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *In Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2004.

[9] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov. Semeval-2016 task 4: Sentiment analysis in twitter. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)*, pp. 1–18, 2016.

[10] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pp. 61–74. MIT Press, 1999.

[11] D. Tang, B. Qin, and T. Liu. Deep learning for sentiment analysis: Successful approaches and future challenges. *Wiley Int. Rev. Data Min. and Knowl. Disc.*, 5(6):292–303, Nov. 2015. doi: 10.1002/widm.1171

[12] D. Tang, B. Qin, and T. Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1422–1432. Association for Computational Linguistics, 2015. doi: 10.18653/v1/D15-1167

[13] S. Xu, H. Liang, and T. Baldwin. Unimelb at semeval-2016 tasks 4a and 4b: An ensemble of neural networks and a word2vec based model for sentiment classification. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pp. 183–189, 2016.

[14] L. Zhang, S. Wang, and B. Liu. Deep learning for sentiment analysis : A survey. *CoRR*, abs/1801.07883, 2018.