Sentiment Analysis using Deep Learning
Review 3

Adriana Ontiveros Gonzalez and Sima Shafaei

Abstract: Sentiment analysis is one of the most active areas of research in AI and Natural Language processing. Nowadays, we can collect a huge volume of data from the internet that is useful not only for individuals but also for organizations. In this paper, we used a deep learning approach and create a Bidirectional LSTM model to classify Amazon reviews on the product category of Kindle Store to positive or negative classes. Obtained results show that this model can achieve 94% accuracy on test data and only 6% of the samples for each class were misclassified.

Keyword: sentiment analysis, deep learning, LSTM, Bidirectional LSTM.

I. INTRODUCTION

Sentiment analysis is the computational study of people's opinions, sentiments, emotions, appraisals, and attitudes towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes (Liu, 2015). On one hand, studies in sentiment analysis can be divided into three general categories: document level, sentence level, and aspect level (Zhang, Wang, & Liu, 2018), and on the other hand, research in this area can be divided into two categories. The first group works on designing effective features for building a powerful sentiment classifier such as word n-grams, text topic, bag-of-opinions, syntactic relations, and sentiment lexicon features. The second set of studies use machine learning methods in a supervised learning framework and is concerned with improving different learning methods. Since the subject of this project is document-level sentiment analysis using a deep learning method, in the following, we briefly introduce several state-of-the-art articles in this area.

Most of the works in this area, improve the architecture of LSTM or CNN for the sentiment analysis application. When dealing with considerably long texts, LSTM fails on capturing and understanding significant sentiment messages. To solve this problem (Xu, Chen, Qiu, & Huang, 2016) proposes a cached long short-term memory neural network (CLSTM) to capture information in a longer step by introducing a cache mechanism. Another limitation of the standard LSTM architectures is that they only allow for strictly sequential information propagation. In (Tai, Socher, & Manning, 2015) authors introduce a generalization of the standard LSTM architecture to tree-structured network topologies and show its superiority for representing sentence meaning over a sequential LSTM. (Tang, Qin, & Liu, 2015) Address the challenge of encoding the intrinsic relations between sentences in the semantic meaning of a document. The model first learns sentence representation with convolutional neural network or long short-term memory. Afterward, the semantics of sentences and their relations are adaptively encoded in document representation with gated recurrent neural networks. Document representations are then used as features for document-level sentiment classification. In (Zhai & Zhang, 2016), authors investigate the usage of auto-encoders in modeling textual data. To address problems of traditional auto-encoders which suffers from scalability with the high dimensionality of vocabulary size and dealing with task-irrelevant words, this paper learns a task-specific representation of the textual data by relaxing the loss function in the auto-encoder to the Bregman divergence and also derives a discriminative loss function from the label information. In particular, they first train a linear classifier on the labeled data, then define a loss for the auto-encoder with the weights learned from the linear classifier. This model is able to take advantage of unlabeled dataset and get improved performance.

The remaining of this paper is organized as follows. Section 2 describes the current project case study, dataset, and data cleaning process. Section 3 presents the selected model, metrics, and methods, Section 4 provides visual presentation and description of results. Finally, in section 5, we conclude the report with some final remarks and further work

II. CASE STUDY DESCRIPTION

Sentiment classification at the document level is to assign an overall sentiment orientation/polarity to an opinion document. In this setting, it is a binary classification task. It can also be formulated as a regression task, for example, to infer an overall rating score from 1 to 5 stars for the review. Some researchers also treat this as a five-class classification task. (Zhang et al., 2018) In this project, we defined sentiment analysis as a binary classification task to determine the polarity on different reviews. We used Amazon reviews in the range of May 1996 to October 2018 provided by the University of California San Diego (UCSD), divided by product categories. the product category of Kindle Store was selected to work on, and just a small subset of the original data was obtained to be the dataset of the project.

This dataset contains 10 features, we kept only the features "reviewText" and "overall." The ratings in the overall feature go from 1, which is the worst rating to 5, which is the best rating. For the purpose of defining

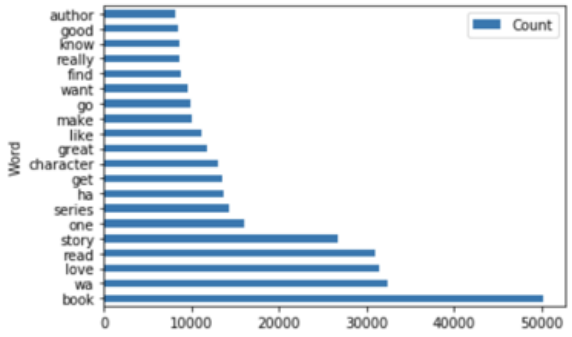

FIG. 1. Horizontal histogram with 20 most frequent words from positive reviews.

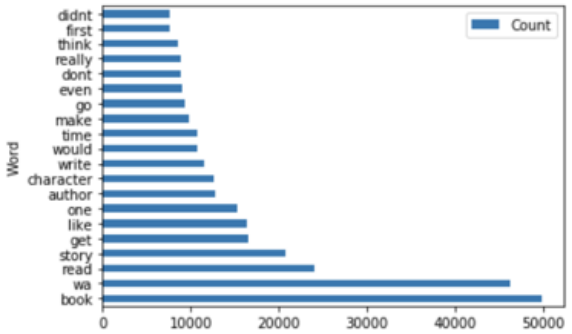

FIG. 2. Horizontal histogram with 20 most frequent words from negative reviews.

which is a positive review and which is a negative one, only the ratings 1 for negative reviews and 5 for positive reviews were kept.

For cleaning the dataset in the first step we removed all samples with the empty "reviewText" feature. Then, we applied random down-sampling to reduce the size of the majority class. and we delete all the samples with reviews written in a language different from English, leaving 46,817 for class 1 and 46,372 for class 0. The complete and cleaned set was split to get from it the 10% for testing, and the remaining 90%, was split on 90% from it for the training and the remaining 10% was for the validation. Afterward, some cleaning on the text of the reviews (on the training and testing datasets separately) was applied to be able to create a bag of words. The text was Unicode, then all was converted to lowercase, the punctuation and digits were removed, also the stemming was applied to all the words, and the stop-words were removed. Next, we evaluate the horizontal histogram with the count of the 20 most frequent words from the positive and negative reviews respectively shown in Fig. 1 and Fig. 2. The most common word for both of the positive and negative reviews is book. We eliminated this word from the dictionary, as the frequency in both of them is the same and this could be noise for the model. Finally, we tokenized the words included in the reviews

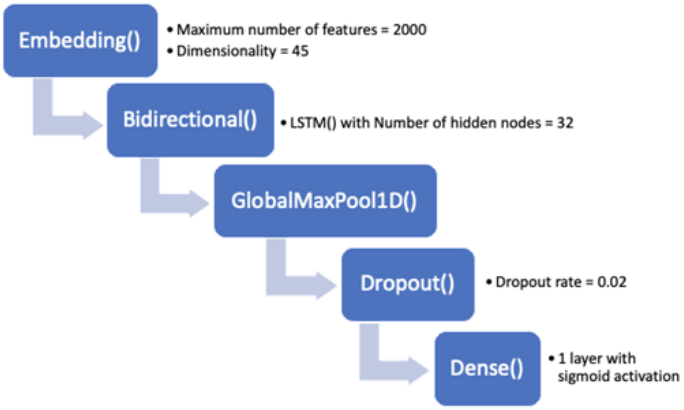

FIG. 3. The architectural model used.

using Keras for the 2,000 most common words included in the training reviews. The training reviews which were strings were then converted into lists of integer indices. This same fit on the training reviews was used to convert the validation and the test reviews into a list of integer indices.

III. SOLUTION

Horizontal histogram with 20 most The model's architecture is shown in Fig. 3. Python was the programming language, the libraries included were pandas, sklearn, tensorflow, and matplotlib.

'Sequential' was the network type, a linear stack of layers. The 'Embedding' layer created a dictionary by mapping the integer indices created during the tokenizing to dense vectors, 2,000 (most frequent words) was the maximum number words index, and 45 was dimensionality based on the average length of the reviews from the training dataset.

Based on related works a 'Bidirectional' Long Short-Term Memory (Bidirectional LSTM) with 32 hidden nodes was used. LSTM modeled more accurate temporal sequences with long-range dependencies. Bidirectional LSTM split into two directions (forward states for positive time direction and backward states for negative time direction) the neurons of a regular LSTM, the inputs of the opposite direction states are not connected to the output. When the two-time directions are being used, the information from the past and future of the current time frame can be used (Pal, Ghosh, & Nag, 2018).

'GlobalMaxPool1D' was used for temporal data, as the total number of parameters is reduced, we minimize the overfitting (Lin, Chen, Yan, 2014). Its layers reduced the spatial dimensions of a 3D tensor, based on the input of the max-pooling layer, the global max-pooling output is the maximum amount in the max-pooling layer. The

dropout was set to 0.02; adding at the end one layer with the sigmoid activation function.

Grid search with cross-validation was used on the training dataset to tune the hyperparameters; to reduce the processing time, two to three options were selected for each parameter. The hyperparameters validated were the following:

- Number of hidden units for the LSTM: The dimensionality of the output layer refers to the state output from an RNN cell (TensorFlow, 2020). 8, 18, and 32 hidden nodes were used in the selection, these were placed in an array to run them in a "for loop," and the grid search was inside this "for loop."
- Dropout rate: To prevent overfitting, works by randomly dropping out neurons during every epoch (TensorFlow, 2020). It should be between 0 and 1, and two small options were selected and place in the grid search, 0.02 and 0.05.
- Number of Epochs: An epoch is one full training cycle on the given set, 3 and 5 were selected, to be placed in the grid search.
- Batch size: Provides the number of training samples to work through before the internal parameters of the model are updated. 8 and 16 were placed in the grid search.

The optimizer was Adaptive Moment Estimation (Adam), it has demonstrated to work well in practice, and it is also favorably compared to other optimization methods (Kingma Ba, 2014). Binary cross-entropy was used for the loss, calculates the cross-entropy loss between the predicted labels and the true ones. The accuracy was selected as a metric.

The cross-validation of the grid search was set to 3-folds, and once the "for loop" containing the grid search finished, the best score obtained, which indicates the mean cross-validated score of the best parameter, was 0.9316, and the best parameters obtained were 32 hidden units, 3 epochs, with a batch size of 8 and dropout rate of 0.02. The architectural model was created with these tuned hyperparameters, then compiled and fit with the training dataset and finally validated with the validation dataset.

IV. RESULTS

Results are shown in Fig. 4, Fig. 5 and Fig. 6. The accuracy of the model kept increasing on every epoch and its loss decreasing, although the difference between the second and third epoch was minimal.

The fitted model was used to predict the test dataset obtaining 94% accuracy, the dataset is balanced, so we can rely on the result. Observing the confusion matrix in Figure, all the numbers on the diagonals look similar

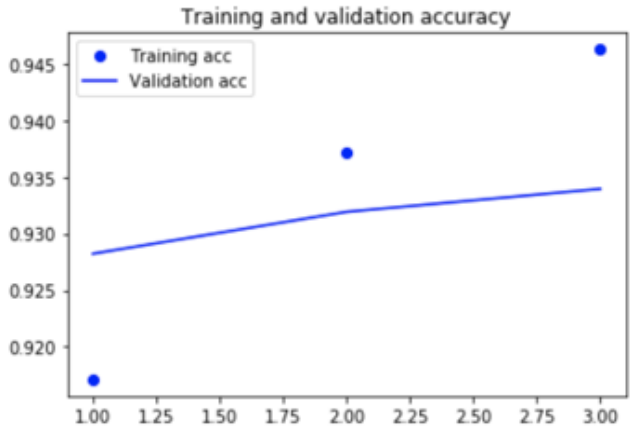

FIG. 4. Training and validation accuracy

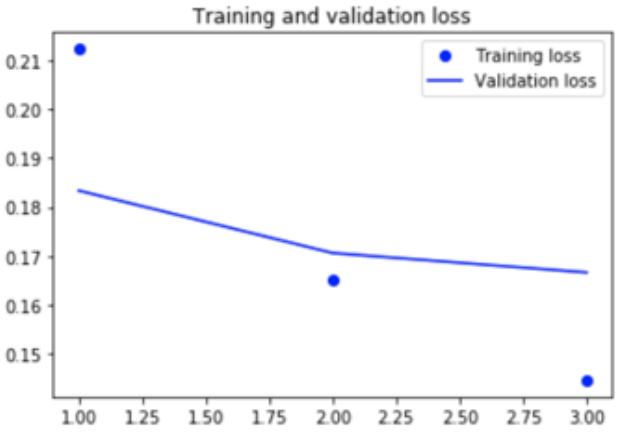

FIG. 5. Training and validation loss

and comparing the true labels and the predicted labels for both classes, only the 6% of the samples for each of them were mispredicted.

V. CONCLUSIONS

In this work, we used a Bidirectional LSTM model to determine the polariry of Amazon reviews. Results shows

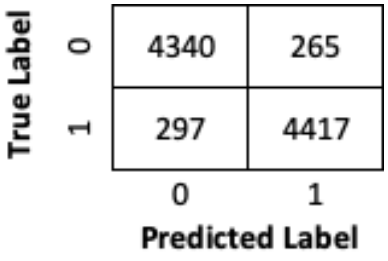

FIG. 6. confusion matrix

that this model can achieve an acceptable accuracy (94%) on test data and only 6% of the samples for each positive and negative class were mispredicted.

One improvement for this paper can be comparing this Bidirectional LSTM model to other existing models on a same dataset which we gave up due to time and resource limitation.

Moreover, there are several popular problems of sentiment analysis classification including sarcasm, negations, word ambiguity, and multi-polarity. avoiding these possible problems and can significantly increase sentiment analysis accuracy in a classification model. There are different solutions to overcome these problems. We can change the structure of the network and use a more complex and deep network to learn sarcasm, negations, word ambiguity, and multi-polarity. Although we used word embedding layer in network architecture to capture context of a word in a document, semantic and syntactic similarity and relation with other words but it is not enough for above sentiment analysis problems.

VI. REFERENCES

Kingma, D. P., Ba, J. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

Lin, M., Chen, Q., Yan, S. (2014). Network in Network.

Liu, B. (2015). Sentiment analysis: Mining opinions, sentiments, and emotions. In Sentiment Analysis: Mining Opinions, Sentiments, and Emotions. https://doi.org/10.1017/CBO9781139084789

Pal, S., Ghosh, S., Nag, A. (2018). Sentiment Analysis in the Light of LSTM Recurrent Neural Networks. International Conference on Information Technology and Applied Mathematics.

Tai, K. S., Socher, R., Manning, C. D. (2015). Improved semantic representations from tree-structured long short-Term memory networks. ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference. https://doi.org/10.3115/v1/p15-1150

Tang, D., Qin, B., Liu, T. (2015). Document modeling with gated recurrent neural network for sentiment classification. Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing. https://doi.org/10.18653/v1/d15-1167

TensorFlow. (2020, 3 7). tf.keras.layers.Dropout. Retrieved from TensorFlow: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dropout

TensorFlow. (2020, 3 7). tf.keras.layers.LSTM. Retrieved from TensorFlow: https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

Xu, J., Chen, D., Qiu, X., Huang, X. (2016). Cached long short-term memory neural networks for document-level sentiment classification. EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings. https://doi.org/10.18653/v1/d16-1172

Zhai, S., Zhang, Z. (2016). Semisupervised autoencoder for sentiment analysis. 30th AAAI Conference on Artificial Intelligence, AAAI 2016

Zhang, L., Wang, S., Liu, B. (2018). Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. https://doi.org/10.1002/widm.1253