Movie Review Sentiment Analysis

Xiwen Zhang, Dongru Jia

Abstract

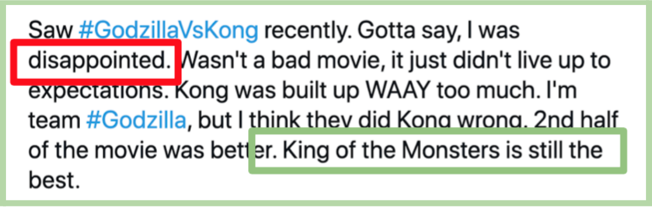
Sentiment Analysis is an essential part of Natural Language Processing, it helps us to identify the sentiment among the text, whether it is a review, comment, or tweet. In this project, Sentiment Analysis is applied to predict the sentiment of each movie review by 'reading' the review text. We approached this problem by implementing multiple machine learning and deep learning algorithms as well as experimenting various word embedding methods. The models included in this project are Logistic Regression, Support Vector Machine, and Convolutional Neural Network. The word embedding methods applied are Bag of Words, Tf-idf, Word2Vec, and a self-trained word embedding by the CNN model.

1. Objective

In this project, we applied multiple algorithms to predict the sentiment of each movie review. However, the purpose of this study is not to find the single best model or word embedding but to discuss the performances of each model with different parameters and the effect of combining deep word embeddings with machine learning models. Afterall, each task and each dataset is different and has its own characteristics, the single best method may only be applicable to a specific case.

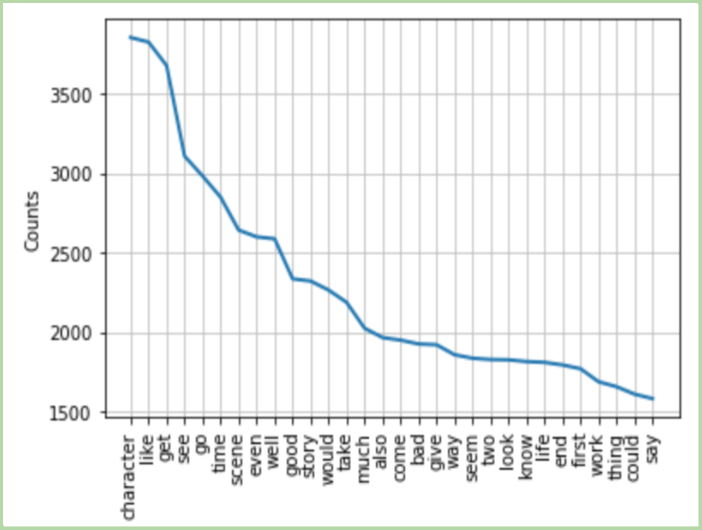
2. Dataset

The dataset we used in this project can be found on Kaggle, whose name and authors are '*Sentiment Polarity Dataset Version 2.0*' and Bo Pang and Lillian Lee. It was distributed with the NLTK book and contains 6 columns and 63652 rows. Most of the columns are just the indexes of the dataset, such as the 'html_id' which indicates the index of the review and 'sent_id' which indicates the index of the sentence within each review. So the most relevant columns to our project are the 'html_id', 'text', and 'tag' which is the label. Each row of the dataset is a single sentence in each review and there are in total 2000 full reviews. If a review is marked positive, then every sentence within that review will all be marked positive. So you may already notice that this could cause some label inconsistency issues, because even if a review favors a movie, it does mean that everything in that review is positive.



Saw #GodzillaVsKong recently. Gotta say, I was disappointed. Wasn't a bad movie, it just didn't live up to expectations. Kong was built up WAAY too much. I'm team #Godzilla, but I think they did Kong wrong. 2nd half of the movie was better. King of the Monsters is still the best.

For example, obviously the review here is overall negative towards the movie; however, the last sentence, 'King of the Monsters is still the best', expresses a very strong positive sentiment. Following the logic of how the dataset is designed, the last sentence will be labeled as negative, which could confuse the models and have a negative effect on the accuracy. How we tackled this challenge is by aggregating the data to review level instead of sentence level. So, the dataset that is fitted to the models contains 2000 rows. Each row is a full review with a maximum length of 80 and average length of 20. Below is an overview of the dataset and some explorations.



character
like
see
get
go
time
well
would
scene
story
come
even
good
take
bad
two
thing
much
also
give
life
seem
first
look
know
end
make
every
new
could
work
actor
really
play
people
think
never
want
big
try
many
act
action
love
director
performance
role
little
best
audience
star
say
plot
man
show
year
find
way
become
back
great
still
watch
another
comedy

3. Background

Because of the development of Natural Language Process techniques, many researchers applied various methods to perform sentiment analysis in the past decades; however, most of the researchers focused on comparing different models or various word embedding methods, there were fewer studies paying attention to discuss the performance of combining deep word embeddings with machine learning models.

After doing research, the study that is most similar to our objective is “*Deep learning for sentiment analysis of movie reviews”* [1]*,* which was done by Pouransari, Hadi & Ghili, Saman. The authors explore various natural language processing (NLP) methods to perform sentiment analysis specifically in the movie review area. The paper applied the bag of words, and skip-gram word2vec models followed by various classifiers, including random forest, SVM, and logistic regression. The result is that the Bag of Words with Logistic regression gains the highest Accuracy. The study inspires us a lot, so we decided to design our project based on this structure.

One of the two most important parts of our study is choosing word embedding methods. Among various word embedding techniques, word2vec, Bag of Words, TF-IDF are used to compare and analyze the performance of movie review sentiment analysis by Park, Hoyeon and Kim, Kyoung-jae[2]. The result of this study is that the performance of TF-IDF was superior to that of Bag of Word and Word2Vec. Their conclusion is a little different from the study by Pouransari, H., and Ghili, S.

The other important part is choosing deep learning models. The paper "*Comparison of Naïve Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification."* [3] sheds the light on the best machine learning approaches to text classification. The authors Pranckevičius, Tomas, and Marcinkevičius, Virginijus present the accuracy for multiple machine learning models, among which the Logistic Regression Classifier outperformed all the rest. The way they compare the deep learning models and how to further optimize the model impressed us in our project.

Besides the models and embedding methods we already studied in the class, there are many new deep learning methods.

Convolution Neural Network is the one that we are interested in. In the paper "*Convolutional Neural Networks for Sentence Classification."* [4], the author presents a novel way, at that time, of classifying text using a Convolutional Neural Network which was heavily used in image processing. This paper proposed a structure that is similar to the tri-gram, 4-gram, and 5-grams BOW model with a much smaller feature set by taking advantage of the deep learning's word embedding method.

Therefore, combining the research of the three studies above, we finally decided to include three models (Logistic Regression, Support Vector Machine and Convolution Neural Network) with four embedding methods (Bag of Words, TF-IDF, Word2Vec and Trained Convolutional Neural Network embedding matrix) in our project.

4. Methodology

4.1 Data Preprocessing

During the preprocessing process, we applied multiple common Natural Language Processing techniques. First, we removed all the characters that are not word characters, so that all the links and series numbers that only show up very few times are cleared. Second, we converted all characters to lowercase and brought them back to their root forms, for example, '*brought -> bring'* and *'numbers -> number'*. Lastly, we removed all the stopwords in English, so that the model can focus on the words that actually matter to determine the sentiment. Another two reasons why we applied these three preprocessing strategies are that it makes sense to use a single token to represent all its different forms semantically and we can also control the size of the feature set which could cause some computational inefficiency issues.

4.2 Models and Embeddings

In this project, we include three types of model, Logistic Regression, Support Vector Machine and a deep learning method, Convolution Neural Network with four different embedding methods. In the two methods that involve word level vectors, we also experimented three different ways of aggregating the embeddings to sentence level.

The project can be splitted into four sections by the four embedding methods. In the first two sections of Bag of Words and Tf-Idf, we applied Logistic Regression and Support Vector Machine with the embeddings. In the third section of Word2Vec, we fit the embeddings to both Logistic Regression and Support Vector Machine by taking the mean, sum and max. Similar to what we did in the previous section, after we retrieved the embedding matrix from the Convolutional Neural Network, we fit the embedding weights to Logistic Regression and Support Vector Machine by following the same steps. So, in total there are 16 models trained, fine-tuned and compared.

5. Results

Since our dataset is perfectly balanced, 1000 positive reviews and 1000 negative reviews, the metric we used to evaluate the results is the accuracy.

Models	**Accuracy**
Logistic Regression w/ embeddings:	
- Bag of Words	*0.7925*
- Tf-Idf	***0.8250***
- W2V (mean)	*0.8050*
- W2V (sum)	*0.8150*
- W2V (max)	*0.6550*
- CNN (mean)	*0.7050*
- CNN (sum)	*0.7050*
- CNN (max)	*0.6850*
Support Vector Machine w/ embeddings:	
- Bag of Words	*0.7400*
- Tf-Idf	*0.8075*
- W2V (mean)	*0.8000*
- W2V (sum)	*0.7875*
- W2V (max)	*0.6400*
- CNN (mean)	*0.4500*
- CNN (sum)	*0.6775*
- CNN (max)	*0.6950*
Convolutional Neural Network	***0.8425***

As shown in the above table, the Convolutional Neural Network model achieved the highest accuracy at 84.25%, which is expected because deep learning models are more flexible and computationally expensive to be trained. However, we noticed that the difference is not very significant, only by 2% compared with the highest accuracy from the rest of the models. We also noticed that Logistic Regression models perform generally better than Support Vector Machine models. All accuracies from Logistic Regression are higher than those from Support Vector Machine except with the CNN embeddings by taking the max.

An interesting fact we observed is that for the Bag of Words and Tf-Idf models, the accuracy is positively correlated with the size of the feature sets. The more words are included, the higher the accuracy goes. However, on the contrary, the highest accuracy of the deep learning models does not always occur when including all the words in the training set. In fact, the optimal number for feature set size, based on this project’s experience, always occurs when less words are included. We suspect that this could result from only taking the word vectors that are sufficiently trained. To be specific, if a word only shows up once in the entire training set, then the weights associated with that word have only gone through very few rounds of gradient descent and are mostly randomly generated. And hence, its word embeddings could only provide very little information when making predictions if not all and bring negative effects.

However, as mentioned in the beginning, identifying the single best model or embedding is not the focus of this project. We are more interested in seeing if the word embedding trained from deep learning models improves the performance of

Logistic Regression and Support Vector Machine. By comparing the accuracies, we can conclude that at least in our case the word embeddings trained by deep learning models contribute little when applied to Logistic Regression and Support Vector Machine models as features. Furthermore, the word embeddings trained from our own Convolutional Neural Network actually have a negative effect on making the prediction accurately and far inferior to the Word2Vec embeddings. One reason we assumed is that the Word2Vec model generates the embeddings by predicting a word when given its context. So the word embeddings trained this way take into account a lot more semantically contextual information than the CNN embeddings which only care about being positive or negative.

6. Discussion

Although we could not validate that the embeddings from deep learning models improve the performance of smaller and faster machine learning models such as Logistic Regression and Support Vector Machine, we strongly believe that much more work could be done before coming to a firm conclusion. For example, we believe that tuning the parameters could certainly improve the quality of the resulting embeddings. Secondly, future work could experiment on aggregating the word embeddings differently and more creatively when fitting them to a machine learning model. Thirdly, performing some feature selection and feature engineering work before fitting could also lead to better performances.

References:

[1] Pouransari, H., & Ghili, S. (n.d.). *Deep learning for sentiment analysis of movie reviews*. Retrieved from https://cs224d.stanford.edu/reports/PouransariHadi.pdf

[2] Park, H., & Kim, K. (2020, August) . *Impact of Word Embedding Methods on Performance of Sentiment Analysis with Machine Learning Techniques.* Retrieved from https://www.koreascience.or.kr/article/JAKO202024758671657.pdf

[3] Pranckevičius, T., & Marcinkevičius, V. (n,d.). *Comparison of Naïve Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification.*Retrieved from https://www.bjmc.lu.lv/fileadmin/user_upload/lu_portal/projekti/bjmc/Contents/5_2_05_Pranckevicius.pdf

[4] Kim, Y. (n,d.). *Convolutional Neural Networks for Sentence Classification.* Retrieved from https://arxiv.org/pdf/1408.5882.pdf