Large Movie Review Data Sentiment Analysis

Hongyuan Zhang, Qi Chu, Xiangyu Zhuang, Xinshu Jiang
Rady School of Business Management, University of California, San Diego
{hoz010@ucsd.edu, qchu@ucsd.edu, xzhuang@ucsd.edu, xij002@ucsd.edu}

Abstract

Sentiment analysis (also known as opinion mining) is the use of natural language processing as well as text analysis to systematically identify, extract and quantify subjective information. There has been a rapid increase in the application of it within areas of social media listening, survey responses analysis, corporate people management, etc. In this work, we perform sentiment analysis on unstructured polarized review data for accurate classification. We also utilize pre-processing techniques and various feature vectors to understand performance differences between models and possible reasons behind it.

1 Introduction

Internet Movie Database (IMDB) is a digital platform that contains vast amounts of information related to films, television series, home videos, video games, and streaming content online – including cast, ratings, and critical reviews etc. Users not only post original reviews but also read other's reviews to see if they will like certain movies. Our group is especially interested in understanding the sentiment embedded in review data. Beyond the sake of personal interest, sentiment analysis of online review also has important applications in real business settings. For example, the like recommendation of clothes in Amazon based on the sentiment analysis of user review. Social media platforms utilize consumer stories and opinion data for advertising, marketing, recommending new friend relationships etc.

In this paper, we are going to utilize various natural language processing and sentiment analysis techniques to do a positive versus negative binary classification to understand the feeling of a user about a specific movie. Various pre-processing methods as well as parameter tuning are included in this process to achieve a better model performance.

2 Literature Review

2.1 Overview of Sentiment Analysis

Sentiment analysis (SA), sometimes referred to opinion mining as well, is a subcategory of natural language processing (NLP). Although opinion mining focuses more on retrieving individuals' opinions towards certain targets and sentiment analysis is more about mining the polarity of people's explicit and implicit attitudes, the meanings are interchangeable in most scenarios (Medhat et al., 1981). Generally, there are three levels of sentiment analysis: document-level, sentence-level and aspect level. Document-level sentiment analysis uses the entire document as input and seeks to classify the sentiment. It assumes that the document expresses one core idea towards a topic, thus it is the most basic and simplest level of sentiment analysis. Sentence-level analysis takes one step further to assume one document has multiple opinions towards even the same entities. It breaks down the analysis into sentence-level and assumes one sentence expresses a single opinion. The first step of sentence-level sentiment analysis is to determine whether the sentence is either subjective or objective before it takes subjective ones to further analysis. Compared with the previous two, aspect-level sentiment analysis is the most fine-grained type. In product reviews, one of the most common applications of sentiment analysis, people comment on products from different perspectives. To precisely capture the valuable information, it requires scientists to first identify the entities and aspects and then classify the corresponding opinions (Feldman, 2013).

Though different levels of analysis differ slightly, sentiment analysis follows a fundamental framework. It is considered as a sequential work of

sentiment identification, feature extraction, and selection, sentiment classification. (Medhat et al., 1981). Sentiment identification refers to the labeling work of the corpus. Feature extraction and selection aim to efficiently transform the raw text input into a vector space that can be recognized by algorithms. Common text features and text representations include Term's presence and frequency, Part-of-speech (POS) information, sentiment words and phrases, rules of opinions, syntactic dependency and sentiment shifters(Liu, 2012; Pang et al., 2008). The sentiment classification techniques can be roughly categorized into machine learning approaches and lexicon-based approaches. Machine learning approaches adopt linguistic features and machine learning algorithms and lexicon-based approaches rely on a sentiment lexicon, a collection of known and precompiled sentiment terms(Medhat et al., 1981).

2.2 Related Work on IMDB Dataset

Convolutional neural network (CNN) is a neural network that can leverage the internal structure of data. In their paper, Rie and Tong utilized CNN to exploit the 1D structure of review text for accurate topic classification. In addition to applying CNN to high-dimensional text data, they also deployed a bag-of-word conversion in the convolution layer. A combined multiple convolution layer is also experimented for better model performance. These experiments demonstrate the effectiveness of the approach in comparison with state-of-the-art methods (Johnson et al.,2015).

Long short-term memory (LSTM) has been successfully used to model sequential data of variable length. In their work, Gui et al introduced a dynamic skip connection, which can learn to directly connect two dependent words and alleviate the inherent weakness of the standard LSTM model. They proposed the use of reinforcement learning to learn the dependent relationship through and this approach helps the better modeling of dependencies within variable length (Gui et al., 2018). The enhanced model outperforms Skip LSTM and Jump LSTM models and achieved 0.91 accuracy on the binary classification. The proposed model not only can achieve sequence modeling tasks such as language modeling and named entity recognition, but it also has a stronger ability for text classification tasks than Jump LSTM and Skip LSTM (Gui et al., 2018).

Furthermore, Li pointed out in his study on the IMDB dataset that Logistic Regression appeared as the best performing baseline algorithm compared to the others such as SMO, Naive Bayes, and J48 (Li, 2019). Here, SMO stands for Sequential minimal optimization that trains on the support vector machine. J48 is an implementation of decision tree algorithm in LightSIDE that when training on the dataset, the algorithm ignores missing values and uses attribute values for the other records as prediction. Li also suggested the Continuous Bag-of-Words (CBOW) model or skip-gram model for word representations, and feature extraction models from blocks of data such as Recurrent Neural Network, CNN, etc for enhanced prediction choices. Based on this study, our team used Logistic Regression as a starting point of our model prediction, and then implemented the Neural Network model for better understanding of the problem.

3 Hypothesis

We assume that BERT and POS tagging with term frequency for adjective sand adverbs will outperform other methods in this movie review classification problem due to the following reasons.

For BERT, it has two main advantages. Firstly, it has already been pre-trained on huge corpus including Wikipedia and Book. Therefore, it can combine both global and local features of words. Secondly, it utilizes both masked language model and next sentence prediction so that the model can have a deeper understanding or words and sentences in certain corpus than other models.

For POS tagging, since sentiments are very likely to be expressed using adjectives and adverbs, term frequency of these words can be of great significance in sentiment prediction. Hence, we believe combing term frequency and POS tagging will be very helpful for this project.

4 Methodology

4.1 Pre-processing

Firstly, we clean the review texts by removing stop words, punctuation and anything that is not alphabet or numbers.

Then, using spaCy (Bird et al., 2009), we can extract tokens from each review and convert all of

them to lower case. We also do lemmatization to switch any kind of a word to its base root. Finally, we are able to build vocabulary with word counts from the processed tokens. For BERT, the text preprocessing is directly completed using the preprocessor API[1] that will be explained in more detail in the experiment section.

4.2 Feature Extraction and Selection

4.2.1 Term-frequency

In binary representation, the feature is a vector of dimension equal to the size of the vocabulary. Values within the vector are either 0 or 1. The value at an index is 1 if the word corresponding to that index is present in the document, else 0.

Term Frequency (TF) refers to how frequently a word occurs in a document. There are words in a document that occur many times but maybe less helpful in understanding the sentiment, for example words like "the", "is", "of", and so forth. When conducting feature engineering, we choose to remove a list of stop words before analysis. However, this may not be a very sophisticated approach to adjusting term frequency for commonly used words. A term's inverse document frequency (IDF) decreases the weight for commonly used words and increases the weight for words that are not used very much in a collection of documents. It is calculated as follows where t is the term (word) we are looking to measure the commonness of and N is the number of documents (d) in the corpus (D).

$$idf(t, D) = \log(\frac{N}{count(d \in D: t \in)})$$

TF-IDF stands for term frequency-inverse document frequency and it represents the frequency of a term adjusted for how rarely it is used. It is the product of TF and IDF score. The higher the TF-IDF score the more important or relevant the term is; as a term gets less relevant, its TF-IDF score will approach 0.

4.2.2 Embedding

Embedding is a kind of text representation that extracts the meaning of words from the corpus. It can learn both syntactic and semantic meanings of words from the contexts in which they are used. The concept of distributional hypothesis suggests that words occurring in similar contexts are semantically similar (Sahlgren, 2006). Each word is converted to a N-dimensional vector and semantic relationships of words are reflected in the directions and distance between vectors.

There are two most popular ways of word embeddings, which are word2vec and glove. Word2vec focuses on word co-occurrence within local domain-related contexts while glove leveraged global word co-occurrence on the entire global corpus. For this project, we focus more on texts that are all about movie reviews, so word2vec is a better choice. As for word2vec, there are two ways for training, namely Skip-gram and CBOW (Continuous Bag of Words). Skip-gram predicts the contexts of each word while CBOW predicts the original word given the contexts (Debo et al., 2018). We will experiment with different settings of word2vec models, including vector size, context window size, and negative sampling and discuss their effects.

4.2.3 Part-of-speech (POS)

Part-of-speech (POS) is a commonly adopted tagging method in the field of phrase mining, given its ability to disambiguate word sense (Wilks et al., 1998). It assigns the grammatical category to each word and helps machines understand the linguistic meaning. Related research has shown that certain grammatical categories (e.g. adjectives, adverbs) are good indicators of subjective opinions (Hatzivassiloglou et al., 2000). However, this does imply other grammatical categories do not contribute to the expression of sentiment (Pang et al., 2008). Given these findings, it's reasonable to assume that, compared with traditional topic classification, the presence of strong subjective opinion indicators is of great importance in sentiment analysis (Liu, 2012). Hence, we would combine the POS tagging and term-frequency techniques to tackle the problem of document-level sentiment analysis and report the performance.

4.2.4 Language Model

Bidirectional Encoder Representations from Transformers (BERT) is a language representation model that pre-trained unlabeled text into deep bidirectional representations by jointly conditioning on both left and right context in all layers (Devlin et al., 2019). The large unlabeled text corpus BERT is trained on are Wikipedia

[1]https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3

(2,500 million words) and Book Corpus (800 million words).

In specific, both Masked Language Model (Masked LM) and Next Sentence Prediction (NSP) are included as a part of the tasks in the pre-training section that allows deep understanding of words and sentences in certain corpus (Devlin et al., 2019). Here, Masked LM refers to the process of masking some percentage of the input tokens at random, and then predicting those masked tokens to get deep bidirectional representation. And the NSP is a vital downstream process that understands sentence relationships. With the corpus and tasks mentioned above, the pre-trained representations are able to reduce the need for many heavily-engineered task specific architectures.

Moreover, fine-tuning is also a part of BERT, where single text or text pairs are swapped out with appropriate inputs and outputs in downstream tasks (Devlin et al., 2019). With the fine-tune ability, the task specific models can benefit from the larger, more expressive pre-trained representations even when downstream task data is very small. With all above mentioned, we acknowledge the empirical power of BERT.

4.3 Sentiment Classification

4.3.1 Lexicon-based Approach

Generally, the lexicon-based model is robust but time-consuming and labor-intensive. VADER (for Valence Aware Dictionary for sentiment Reasoning) is a parsimonious rule-based model using a combination of qualitative and quantitative methods (Hutto et al., 2014). It proves to be well-suited for social media contexts as it leverages a self-constructed gold-standard list of lexical features. It will serve as the baseline for our movie reviews sentiment classification.

4.3.2 Machine Learning Approach

We use two approaches for classifying sentiments of movie reviews, which are logistic regression and neural network multilayer perceptron classifier (MLP). For logistic regression, we use grid search cross-validation on the training set to tune hyper-parameters, such as types of penalty (l1, l2) and inverse of regularization strength.

For MLP, we tune hyper-parameters including hidden layer sizes, solver, early stopping, and validation fraction. We use the experience formula for hidden layers:

$$h1 = int(log_2 n)$$
$$h2 = int(\sqrt{m + n})$$
$$h3 = 2 * n + 1$$

where, n = dimension of features, m = dimension of output.

There are three kinds of combinations for hidden layer sizes, (h1), (h2, h1), (h3, h2, h1).

For the POS model, we also utilize XGBoost feature importance to select the most important features and reduce dimensionality. As for model evaluation metrics, we use classification accuracy, AUC score and f1-score.

5 Experiments

5.1 Dataset

We employ the widely used movie review dataset contributed by Mass et al. (2011) to test and validate our assumptions and methodologies. The 50k reviews are evenly split into a 25k train set and a 25k test set. Labels are provided consisting of balanced 50% positive and 50% negative ones without the neutral class. Negative reviews are sampled from reviews with a score <= 4 out of 10. Positive reviews are the ones that score >= 7 out of 10. The average review length in terms of words is 237 for positive samples and 231 for negative samples in the training set. The maximum of the raw corpus is 2,310 and the minimum is 10. Distribution of raw corpus length on the training set can be illustrated as below:

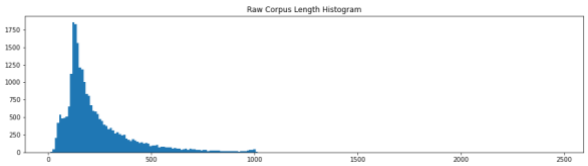

Figure 1: Histogram of raw corpus length on training set

5.2 Model Details

5.2.1 Vader

For VADER, we did no preprocessing and used the raw movie corpus as the input for sentiment classification. The rule for the prediction can be denoted as:

$$prediction = \begin{cases} positive, if\ score_{pos} \geq score_{neg} \\ negative, if\ score_{pos} < score_{neg} \end{cases}$$

, where $score_{pos}$ and $score_{neg}$ is the output of the VADER algorithm.

5.2.2 Term Frequency

Before fitting the model, we first look at frequencies of pre-processed terms appearing in

the training dataset. We notice the frequency distribution has a long tail with. Several words that have the most frequent presence include "the, film, movies "which are not helpful in understanding the review sentiment. Therefore, we experiment various frequency cutoffs and choose to remain terms with frequency between 30 and 25000. This helps develop a refined corpus of 7911 words, which is used to construct features for each individual review. The vector size of each review is 7911. The difference between 3 methods is listed below:

- *Binary-encoding*: the value at an index is 1 if the word corresponding to that index is present in the corpus, else 0.
- *Frequency*-encoding: the value corresponding to each word is its frequency in the document.
- *TF-IDF*-encoding: the value corresponding to each word is its tf-idf value.

5.2.3 Embedding

The following are some of the key hyperparameters settings for word2vec models:

Training Algorithm: Since skip-gram has a better representation for rare words or phrases than CBOW, it is more suitable for our task as we need to capture semantic meanings not just for frequent words, but also those words with low-frequency to have a more accurate sentiment classification. And it has a better performance.

Vector size: As dimensionality increases, quality of word embedding increases. However, the training time also increases a lot and marginal gain diminishes. We try different sizes and find out that vector size of 1000 reaches a good trade-off between embedding quality and required training time.

Context window: Since larger windows tend to capture more topical or domain information while smaller windows tend to capture more about the word itself. For our project, topical information is very important and we find that a context window of 20 is most appropriate.

Negative sampling: By setting a negative sampling of 10, the algorithm only samples 10 negative instances for each word and context pair during optimization. Therefore, it reaches a great tradeoff between computation efficiency and embedding quality.

5.2.4 Part-of-speech (POS)

We first apply a series of preprocessing techniques to extract clean lowercase tokenized data from the raw corpus using spaCy (Bird et al., 2009). The techniques include tokenization, stemming, lemmatization, and stop words and punctuation removal.

For feature extraction and selection, we use the NLTK (Honnibal et al., 2020) for POS tagging to select only adjectives and adverbs as the vocabulary and apply binary encoding to form the feature matrix. The size of all the adjectives and adverbs extracted is 31,974 and is reduced to 10,641 based on words' frequencies. XGBoost is applied to further reduce dimensions of the feature matrix based on feature importance.

We use simple logistic regression and Multi-Layer Perceptron (MLP) as the classifier, as MLP is better at catching the non-linearity present in the feature matrix.

5.2.5 BERT

Prior to the implementation of the BERT model, the dataset was directly loaded using the URL of our study[2]. Using a companion of the BERT model named SavedModel, a preprocessor API is implemented for text embedding with Transformer encoders. The text inputs have been normalized in lowercase, tokenized, and stripped accent markers. Specifically, the encoder input details contain seq_length=128, batch_size=32. A dict of three int32 Tensors, all with the same batch_size and seq_length is then generated. As a result, a batch of fixed-length input sequences for the Transformer encoder is included prepared for further compilation.

The BERT model implemented is based on the TensorFlow Models repository on GitHub. Specifically, the Transformer blocks included in the model are denoted as L=12 hidden layers, a hidden size of H=786, and A=12 attention heads. This is a base BERT model out of the other choices such as small BERT and large BERT. While the small BERT is more time efficient, the results are not as good as the other two. Although the large BERT can generate better performance, it is not computable with our current resource using GPU

[2] https://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz

on Google Colab working space. For training purposes, random input masking has been applied independently to words. A total of three epochs are used in the fine-tuning section with an initial learning rate of 3e-5. Here, both loss and accuracy are taken into consideration and the optimizer selection is Adam.

5.3 Overall Performance

We use accuracy and AUC score to report and compare performances on different methods as we deal with a balanced dataset. The below table show a comparison of different methods:

model	Accu.	f1	Preci.	Rec.	AUC
VADER	0.693	0.739	0.643	0.87	-
binary	0.855	0.855	0.856	0.854	0.927
TF	0.823	0.822	0.825	0.82	0.900
TF-IDF	0.804	0.821	0.756	0.898	0.895
w2v (logit)	0.874	0.872	0.881	0.863	0.942
w2v (MLP)	0.873	0.872	0.88	0.865	0.943
POS (logit)	0.822	0.82	0.83	0.811	0.900
POS (logit_ select)	0.84	0.84	0.844	0.836	0.917
POS (MLP_ select)	0.87	0.87	0.866	0.875	0.941
BERT	0.887	-	-	-	-

* Due to limited resource, other metrics of BERT model are currently unavailable

Table 1: Model Performance

Term-frequency

Model based on binary encoding achieved a f1 score of 0.85 and it also has the best performance among the three frequency-based features. Since we performed manual cutoff based on observed term frequency, this may be the reason that negatively affects the accuracy of models based on term-frequency and TF-IDF vectors.

Embedding

Using features from our final word2vec skip gram model, logistic regression gives a slightly better accuracy and F1-score while MLP classifier has a slightly higher AUC.

Part-of-speech (POS)

We could see from the results that the MLP model without feature selection outperforms others. Overall, MLP models outperforms logistic regression and are less sensitive to the dimensions of the input matrix. The ability to capture the non-linearity matters in this case of classification.

BERT

The BERT model uses a large amount of pre-training corpus such as Wikipedia and Book Corpus. In addition, the Masked LM and NSP training methods ensure the understanding of fitted text. With the additional power of fine-tuning, the BERT model is expected to outperform the other models. Indeed, the accuracy score of the BERT is 0.887.

However, a limitation is that BERT lacks the ability to handle long text sequences. The default value of the maximum token supported by BERT is 512. The maximum sentence length for our dataset is 2,310, however, most of the sentence length is below 500. Although there is not a major effect from the sentences that exceed 512 length, this could lead to suboptimal results.

6 Results and Conclusion

In this paper, we tried various sentiment analysis techniques as well as models such as frequency-based, Word2vec, Part-of-speech and BERT to understand the review data. Based on the selected IMDb dataset, BERT has the best performance with an accuracy of 0.887 on the test set, which indicates a stronger ability to detect the underlying sentiment. Bert possesses a stronger ability to process larger amounts of text and language. The better performance is achieved via transfer learning through using a pre-trained model based on a vast corpus.

7 Discussion

The models we have now include Vader, Term frequency, Word2Vec, Part-of-Speech and BERT. With current understanding of data and models, we can achieve the highest scores of 0.887. In the future, a possible approach to further improve the sentiment level classification would be to use an embedded model that takes account to the best performing classifiers with weights and generate a more accurate prediction based on improved learning of data characteristics.

Additionally, the sentiment level classification we performed in this study could be further expanded to aspect level analysis. For example, the Part-of-Speech model could be combined with CNN or LSTM to achieve better performances.

References

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Deho, B. O., Agangiba, A. W., Aryeh, L. F., & Ansah, A. J. (2018, August). *Sentiment analysis with word embedding*. In 2018 IEEE 7th International Conference on Adaptive Science & Technology (ICAST) (pp. 1-4). IEEE.

Feldman, R. (2013). *Techniques and applications for sentiment analysis*. Communications of the ACM, 56(4), 82-89.

Gui T, Zhang Q, Zhao L, Lin Y, Peng M, Gong J, Huang X. *Long short-term memory with dynamic skip connections*. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33; 2019. p. 6481-88. K.U. Jaseena and B.C. Kovoor

Hatzivassiloglou, V., & Wiebe, J. (2000). *Effects of adjective orientation and gradability on sentence subjectivity*. In COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics.

Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). spaCy: *Industrial-strength Natural Language Processing in Python*. https://doi.org/10.5281/zenodo.1212303

Hutto, C., & Gilbert, E. (2014, May). Vader: *A parsimonious rule-based model for sentiment analysis of social media text*. In Proceedings of the international AAAI conference on web and social media (Vol. 8, No. 1, pp. 216-225).

Jacob, D., Ming-Wei C., Kenton L., & Kristina T. (2019, May). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/arXiv.1810.04805

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011, June). *Learning word vectors for sentiment analysis*. In Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies (pp. 142-150).

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. Ain Shams engineering journal, 5(4), 1093-1113.

Pang, B., & Lee, L. (2008). *Opinion mining and sentiment analysis.* Foundations and Trends® in information retrieval, 2(1–2), 1-135.

Rie Johnson and Tong Zhang. 2014. *Effective use of word order for text categorization with convolutional neural networks*. arXiv preprint arXiv:1412.1058.

Sahlgren, M. (2008). *The distributional hypothesis*. Italian Journal of Disability Studies, 20, 33-53.

Wilks, Y., & Stevenson, M. (1998). *The grammar of sense: Using part-of-speech tags as a first step in semantic disambiguation*. Natural Language Engineering, 4(2), 135-143.