Sentiment Analysis on IMDb Movie Reviews: Word2Vec with CNN & Naive Bayes

Abstract

Movie reviews provide critical analysis and insights into various features of a film, including the plot, cinematography, music, and genres, which allows viewers to make informed decisions on what movies to watch. IMDb (Internet Movie Database) is one of the most popular online databases for movies, television series, and other media source content designed to help users explore the world of movies and shows. Sentiment analysis is an extraction of sentiment, a writer's positive or negative orientation toward some object. It helps us understand the relationship between natural text and human emotions or judgment. For movie review datasets, it can be used to rate how positive or negative a review is and get the overall polarity of a movie. For our project, we apply sentiment analysis on a set of movie reviews using Word2Vec word embeddings and two classifiers: Naive Bayes and Convolutional Neural Networks (CNN). We found that the CNN model passed the evaluation process producing a 0.90 F1 score and a testing accuracy of 90%, outperforming our baseline model, Naive Bayes, which had an F1 score of 0.77 and a testing accuracy of 77.2%. We also found that changes in vector size have an impact on the system's ability to forecast positive and negative feelings effectively.

1. Introduction

IMDb (Internet Movie Database) is one of the most popular online databases for content related to movies, TV shows, actors, directors, and other industry professionals. In 2022, they reported a total of 10.1 million titles in their database, out of which 605,284 titles were movies. It has become an indispensable tool for a wide range of individuals, from film enthusiasts to casual viewers, as a platform to read and share critiques on movies and TV shows. Given the extensive user base and active community and IMDb's substantial usage, it is able to influence and shape public opinions and perceptions of movies. In order to tackle a large dataset like IMDb, particularly movie reviews, there have been developments of various kinds of Machine Learning (ML) and Natural Language Processing (NLP) models approaching the data categorization problem, one of which is sentiment analysis.

Sentiment analysis is an extraction of sentiment, a writer's positive or negative orientation toward some object. It helps us understand the relationship between natural text and human emotions or judgment. Sentiment Analysis on IMDb movie reviews is used to identify the judgment or opinion expressed by a reviewer towards a movie and get the overall polarity of a movie. By analyzing the sentiments expressed in reviews, users can quickly assess whether a movie is well-received or not, which can influence their decision to watch it. It also reduces the complexity of making this prediction.

Our project aims to implement sentiment analysis on IMDb movie reviews using Word2Vec word embeddings paired with two classifier models, Convolutional Neural Networks (CNN) and Naive Bayes to categorize a movie review as either "positive" or "negative". Ultimately, our models should provide information that helps users align their preferences with the sentiments expressed in reviews, aiding them in selecting movies that align with their tastes.

2. Related works

Sentiment analysis is a valuable tool in NLP that systematically classifies affective states of text data, which allows us to establish a relationship between natural text and human

judgment. We discuss several different research studies on sentiment analysis of movie reviews that we draw inspiration from.

There are several findings from prior research studies that show Convolutional Neural Networks have yielded significant outcomes in sentiment analysis tasks. Haque et al. (2019) explore three deep neural network architectures, CNN, LSTM, and LSTM-CNN to classify IMDb movie reviews. The results of their study showed that CNN outperformed all other approaches with a 90% test accuracy and a 91% F1 score. Similar to our approach, Haque et al. (2019) use a Word2Vec embedding layer trained using the IMDb movie review dataset. In line with this research, Mohamed Ali et al. (2021) also applied sentiment analysis over the IMDb movie reviews dataset with deep learning methods including MLP, CNN, LSTM, and a hybrid CNN_LSTM model with word2vec embeddings as well. Although they found that CNN_LSTM yielded higher accuracy, the CNN model followed with a high accuracy of 87.7%. The results of their experiment also suggested that the use of word embedding with deep neural networks effectively yields performance improvements in terms of run time and accuracy. Unlike the research studies discussed, Gandhi et al. (2021) use movie review tweets instead of an IMDb movie review dataset. We will be implementing a research paradigm that follows the CNN architecture Word2Vec modeling from these previous studies. While the aforementioned research studies compare CNN with different deep learning models, our project will focus on the performance analysis of CNN and a baseline model, Naive Bayes using accuracy, precision, recall, and F1-score as our evaluation metrics. We found a comparative analysis between machine learning (improved Naive Bayes) and deep learning (gated CNN or GCNN) approaches for sentiment analysis of movie reviews as seen in the work done by Gowri et al. (2022). The findings show that improved Naive Bayes performed better than GCNN, which is unlike the other work we have explored. They also note that the end result of the Naive Bayes model is accurate, and the process quicker.

Previous studies have also explored the efficacy and accuracy of the Naive Bayes classifier for sentiment analysis of movie reviews (Khan et al., 2021; Samsir et al., 2022). Despite its simplicity, the naïve Bayes‘ classifier has been proven to work satisfactorily in many domains. The approach that Samsir et al. (2022) employ in their study involves using the Word2Vec feature extraction method and the Naïve Bayes classification algorithm, which is parallel to the model we have proposed. An interesting finding from this study was that a Word2Vec embedding with a vector size of 500 yields a high F1-score of 79.77% and 79.44% accuracy as compared to vector sizes 300 and 400. They also found that lemmatization, a preprocessing technique that groups together the inflected forms of the same word like ‘caring’ and ‘care’, decreases the accuracy and F1-score for the sentiment analysis of IMDb movie reviews. As seen in Samsir et al. (2022) we are also interested in exploring the effect of vector sizes on the accuracy and F1-score and we propose to do it for both models, Naive Bayes and CNN. Moreover, using this study, we omitted lemmatization from our data preprocessing step. Similar to this research, Khan et al. (2021) use Word2Vec feature extraction to obtain word vectors and then train different machine learning classifiers Gaussian Naive Bayes, linear support vector classifier, K-nearest neighbors (KNN), logistic regression, and random forest. A Gaussian Naive Bayes classifier performed with 78.01% accuracy with an F1-score of 77.7%. Khan et al. (2021) particularly focus on using a 300-dimension feature vector and their averaging technique to get vectors for each review, which they suggest makes the training and inference processes faster. There are also research studies that do not use Word2Vec embeddings, but utilize the

bag-of-words approach instead for the sentiment analysis of IMDb movie reviews (Dey et al., 2016). This was done on a much smaller dataset of 10,000 movie reviews.

Collectively, these research works give us a ballpark estimate of the evaluation metric values we can predict to see in our project.

3. Data

We used the IMDb dataset containing 50,000 movie reviews created by Andrew Maas at Stanford University (Maas et al., 2011). The dataset contains highly polar movie reviews for training and testing and was designed for binary sentiment classification. We retrieved the CSV (comma-separated values) file from Kaggle, an open-source website that allows users to find and publish datasets.

4. Methodology

4.1 Pre-processing Data

Before fitting the dataset into our models, we cleaned the raw data by using regular expressions to convert all text to lowercase and to remove punctuation marks, HTML tags, URLs, characters that are not letters or digits, and successive whitespaces. We also removed stopwords using the set of 40 stopwords in the English language in the NLTK (Natural Language Toolkit) library. Table 1.1 demonstrates a sample of raw data for both negative and positive sentiments and Table 1.2 demonstrates the same examples after data cleaning:

sentiment	review
Positive	Probably my all-time favorite movie, a story of selflessness, sacrifice and dedication to a noble cause, but it's not preachy or boring. It just never gets old, despite my having seen it some 15 or more times in the last 25 years. Paul Lukas' performance brings tears to my eyes, and Bette Davis, in one of her very few truly sympathetic roles, is a delight. The kids are, as grandma says, more like "dressed-up midgets" than children, but that only makes them more fun to watch. And the mother's slow awakening to what's happening in the world and under her own roof is believable and startling. If I had a dozen thumbs, they'd all be "up" for this movie.
Negative	Besides being boring, the scenes were oppressive and dark. The movie tried to portray some kind of moral, but fell flat with its message. What were the redeeming qualities?? On top of that, I don't think it could make librarians look any more unglamorous than it did.

Table 1.1: positive and negative sentiment movie review samples before cleaning

sentiment	review
Positive	probably time favorite movie story selflessness sacrifice dedication noble cause preachy boring never gets old despite seen 15 times last 25 years paul

	lukas performance brings tears eyes bette davis one truly sympathetic roles delight kids grandma says like dressed midgets children makes fun watch mother slow awakening happening world roof believable startling dozen thumbs movie
Negative	besides boring scenes oppressive dark movie tried portray kind moral fell flat message redeeming qualities top think could make librarians look unglamorous

Table 1.1: positive and negative sentiment movie review samples after cleaning

For the CNN model, we tokenized the text of the dataset with Keras text preprocessing class Tokenizer, which allows us to vectorize text by turning them into a sequence of integers. This is done so that string inputs can be converted into integer inputs suitable for a Keras embedding layer, especially crucial for implementing the CNN model. Lastly, we divided our training and testing set using train_test_split in the Scikit-Learn library into 80% training data and 20% testing data.

4.2 Word2Vec

Word2Vec is one of the most popular methods to generate vector representations, or word embeddings based on the context in which they appear in the text corpus. The algorithm was developed by Tomas Mikolov in 2013 at Google and uses a two-layer neural network: Continuous Bag of Words(CBOW) and Skip-gram. For our study, we implemented word embeddings using the Gensim word2vec model. We trained the word2vec model with the pre-processed IMDb dataset with the following hyperparameters: vector_size=300, window=3, min_count=1, workers=16.

4.3 Sentiment analysis

We built a Naive Bayes Model and Convolutional Neural Network model to conduct sentiment analysis on the movie review dataset and classify the sentiment as either positive or negative.

4.3.1 Naive Bayes

Naive Bayes is a supervised machine learning algorithm widely and commonly used in NLP for sentiment analysis. It is a probabilistic classifier based on the Bayes Theorem i.e. the probability of an event A occurring, based on the prior knowledge of the occurrence of another event B. It is called “naive” because events or features are assumed to be independent of each other. In our study, we chose an NB classifier to be our baseline model. We chose to use the Gaussian Naive Bayes classifier, an extension of Naive Bayes that assumes all the continuous variables associated with each feature to be distributed according to a bell-shaped Gaussian Distribution (also known as the normal distribution). Since it can handle both categorical and continuous features, it makes for a versatile classifier. We build the model using the Gaussian Naive Bayes class in the Scikit-Learn library.

4.3.2 Convolutional Neural Network (CNN)

Convolution Neural Network (CNN) is a class of deep neural networks that is most commonly applied to analyze visual imagery. It has three main layers: convolutional layer, pooling layer, and fully connected (FC) layers. We utilized Keras to build a CNN model for

sentiment analysis. For our model, a Sequential model is defined, which is a linear stack of layers that deals with the ordering of layers within a model. Then we add an embedding layer using the embeddings derived from the Word2Vec-trained model with an embedding space of 300 dimensions. The Convolutional Layer is the core of the CNN architecture that is responsible for the bulk of the computation and feature extraction. The convolutional layer slides a filter over the input, computing the dot product of the weights and the input to create a feature map. In this study, we only use a one-dimension convolutional layer, Conv1D. We created three Conv1D Layers with the following hyperparameters: convolutional layer 1: filter_size = 50, kernel_size = 3; convolutional layer 2: filter_size = 100, kernel_size = 3; convolutional layer 1: filter_size = 50, kernel_size = 3.

Every convolutional layer is followed by a Maxpool Layer and a Dropout Layer. A Maxpool Layer is added to reduce the spatial dimensions (height and weight) of the output feature map generated by the convolutional layer. This is done by calculating the maximum value for each patch of the feature map. A Dropout layer set to 0.2 dropout rate was also added to avoid the overfitting of data. Finally, we applied two activation functions, ReLU (rectified linear unit) and softmax, into the model. Activation functions in the hidden layer provide control over the efficacy of the network model learning a training dataset. Finally, this model was compiled and trained. We fit the training data into our models with the following hyperparameters: batch_size=16, epochs=2. Figure 1 demonstrates a high-level representation of the CNN architecture we implemented for our project:

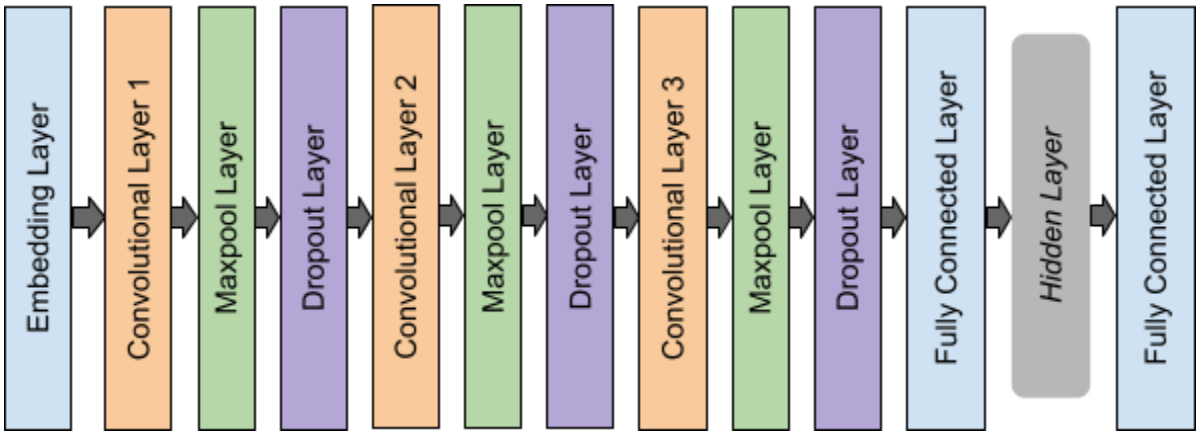

Fig. 1: Higher level representation of CNN architecture

5. Results

The assessment metrics we used to analyze the performance of our models are precision, recall, F1-score, and accuracy. Table 2.1 demonstrates the results of our study:

Evaluation Metrics	**Naive Bayes (NB)**	**Convolutional Neural Network (CNN)**
Precision	0.7662	0.9013

Recall	0.7883	0.9006
F1-Score	0.7771	0.9006
Accuracy	0.7721	0.9006

Table 1: performance analysis of sentiment analysis using Naive Bayes and CNN

As seen in Samsir et al. (2022), Table 2.1 and 2.2 demonstrates the comparative study based on the analyses collected using three different vector sizes for the Naive Bayes and CNN model. The vector size is the number of dimensions of the vector space that gensim Word2Vec maps the words onto.

Vector Size	Naive Bayes			
	Precision	Recall	F1-Score	Accuracy
300	0.7662	0.7883	0.7771	0.7721
400	0.7658	0.7857	0.7770	0.7728
500	0.7706	0.7894	0.7801	0.7756

Table 2.1: Efficacy of Word2Vec size vector between 300, 400, and 500 with Naive Bayes

Vector Size	Convolutional Neural Network (CNN)			
	Precision	Recall	F1-Score	Accuracy
300	0.9013	0.9006	0.9006	0.9006
400	0.8927	0.8906	0.8905	0.8906
500	0.9011	0.8979	0.8977	0.8979

Table 2.2: performance analysis of sentiment analysis using Naive Bayes and CNN

6. Discussion

From the results of our study, the CNN model passed the evaluation process producing a 0.90 F1 score and a testing accuracy of 90%. This model outperformed our baseline model, Naive Bayes, which had an F1 score of 0.77 and a testing accuracy of 77.2%. This performance analysis shows that a deep learning algorithm performs better than a regular machine learning algorithm in the sentiment analysis of IMDb movie reviews. CNN models are generally better at text classification tasks as it utilizes spatial data, hence being able to capture contextual information, while in Naive Bayes words are independent of each other. The CNN model had a high precision rate suggesting its strong efficacy in detecting true positives. In contrast, the Naive Bayes model had a higher recall score compared to its precision score, from which we can infer that it was better at correctly identifying the items present in the input.

Moreover, inspired by Samsir et al. (2022), we compared the effect of different vector sizes on the performance of both the CNN and Naive Bayes models. We kept the window size of 3 and the number of workers at 16 consistent across all the tests. In line with the findings of Samsir et al. (2022), we found that a 500 vector size yields the best performance for the Naive Bayes, compared to vector sizes 300 and 400 as shown in Table 2.1. It is important to note that while there was some performance improvement, it was only a marginal increase. This positive correlation between performance and vector size could be attributed to the fact that the Word2Vec model can capture more information about each word. To reiterate, Word2Vec generates vector representations, or word embeddings based on the context in which they appear in the text corpus. A larger embedding space means more dimensions are available allowing for more complex relationships between words to be captured and reducing sparsity, which can improve the accuracy of the model. It is also possible that a large vector size could generalize better to new, unseen data.

To bridge the gap in research for the effect of vector sizes on the performance of a CNN model on sentiment analysis, we also analyzed how varying vector sizes affect our CNN model, using the same parameters as the one mentioned for Naive Bayes. In contrast to the findings for the Naive Bayes model, the results generated by the CNN model were mixed, as shown in Table 2.2. There is a significant drop in all evaluation metrics, precision, recall, F1 score, and accuracy by around 0.1 when moving from a vector size of 300 to 400. However, there is an increase in all evaluation metrics when the vector size is increased to 500. The results are marginally lower than the results generated by a 300-vector size. Although the cause of this inconsistency remains unclear, the general decrease in accuracy and F1 score could be due to overfitting. It could also be due to increased noise in the embedding space, caused by the increase in parameters for the model to learn. These findings tell us that the changes in vector size have an impact on the system's ability to forecast positive and negative feelings effectively. We find that the Naive Bayes sentiment analysis model benefits from a larger embedding space, but a CNN model does not.

Although all efforts were made to eradicate shortcomings in this study, there are still some limitations. Due to time constraints, we were unable to optimize our CNN model as model training was taking anywhere between 1 hour to 7 hours. In the future, hyperparameter tuning should be done to increase model performance for optimal results. Hyperparameter tuning uses the processing infrastructure of Google Cloud to test out different hyperparameter values while evaluating the performance of the model. This allows researchers to tweak these values to get significant increases in accuracy and other metrics, speed the learning process and reduce overfitting. Future research could also benefit from observing the difference in performance across more algorithms like support vector machine (SVM), random forest, K-Nearest Neighbor (KNN), and other neural networks approaches like LSTM and RNNs.

Project Code

The code for our project can be found on GitHub using the link below:
https://github.com/urvisuwal/sentiment-analysis-using-CNN-and-Naive-Bayes.git

References

Dey L., Chakraborty, S. Biswas, A. Bose, B., & Tiwari S. (2016). Sentiment Analysis of Review Datasets Using Naïve Bayes‘ and K-NN Classifier. International Journal of Information Engineering and Electronic Business, 8(4), 54–62. https://doi.org/10.5815/ijieeb.2016.04.07

Gandhi, U. D., Malarvizhi Kumar, P., Chandra Babu, G., & Karthick, G. (2021). Sentiment Analysis on Twitter Data by Using Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM), *Wireless Personal Communications.* https://doi.org/10.1007/s11277-021-08580-3

Gowri, S., Surendran, R., Divya Bharati, M., & Jabez, J. (2022). Improved Sentimental Analysis to the Movie Reviews using Naive Bayes Classifier. *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, pp. 1831-1836, DOI:10.1109/ICEARS53579.2022.9752408.

Khan, A., Majumdar, D., Mondal, B. (2021). Machine Learning Approach to Sentiment Analysis from Movie Reviews Using Word2Vec. *Proceedings of Research and Applications in Artificial Intelligence*. 131-140 https://doi.org/10.1007/978-981-16-1543-6_12

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2022). Learning Word Vectors for Sentiment Analysis. *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*.

Haque, M. R., Akter Lima, S., & Mishu, S. Z. (2019). Performance Analysis of Different Neural Networks for Sentiment Analysis on IMDb Movie Reviews, *3rd International Conference on Electrical, Computer & Telecommunication Engineering (ICECTE),* Rajshahi, Bangladesh, pp. 161-164, doi: 10.1109/ICECTE48615.2019.9303573.

Mohamed Ali, N., El Hamid, M. M. A., & Youssif, A. (2019). SENTIMENT ANALYSIS FOR MOVIES REVIEWS DATASET USING DEEP LEARNING MODELS. *International Journal of Data Mining & Knowledge Management Process,* 09(3). 19-27. https://doi.org/10.5121/ijdkp.2019.9302

Samsir, K et al.(2022). Implementation Naive Bayes Classification for Sentiment Analysis on Internet Movie Database, *Building of Informatics, Technology and Science,* Volume 4, DOI 10.47065/bits.v4i1.1468

Appendices

Tutorials used to learn about CNN

https://thinkingneuron.com/how-to-classify-text-using-word2vec/

https://www.youtube.com/watch?v=8YsZXTpFRO0&themeRefresh=1