

# PROJECT REPORT

# CSE-6363-003\_004

# Machine Learning

# NLP Based SENTIMENT ANALYSIS FOR MOVIE REVIEWS

**Sai Vineeth, Akula**       - 1002028631

**Yashwanth Rao, Varala Balaji** – 1001916698

**Tharun Reddy, Nalabolu**         - 1002069440

I have neither given nor received unauthorized assistance on this work. I will not post the project description and the solution online.

Sign: Yashwanth Rao Varala Balaji, Tharun Reddy, Nalabolu, Sai Vineeth, Akula

Date: 04/29/2023

Abstract:

Sentiment analysis, a popular application of Natural Language Processing (NLP), involves using machine learning techniques to identify and classify the sentiment expressed in text data, such as customer reviews, social media posts, and product feedback. In this report, we present a project that focuses on building a sentiment analysis model using deep learning algorithms to analyze and classify feedback/reviews from audiences. We use a dataset of 40,000 reviews and preprocess the data to fit into different classification methods. We compare the accuracy of the model across different methods and select the best-performing one. The project aims to provide insights into the performance differences between positive and negative reviews and develop an effective sentiment analysis model for practical applications.

Introduction:

Sentiment analysis has gained significant attention in recent years due to its potential in extracting valuable insights from large amounts of text data. Companies, organizations, and individuals can benefit from understanding the sentiment expressed in customer reviews, social media posts, and other text sources. Sentiment analysis enables businesses to monitor customer feedback, assess product or service performance, make data-driven decisions, and improve customer satisfaction.

In this report, we present a project that focuses on building a sentiment analysis model using deep learning algorithms. We aim to analyze and classify feedback/reviews from audiences as positive or negative based on the sentiment expressed in the text. We use a dataset containing 40,000 reviews and preprocess the data to fit into different types of classification methods. We then compare the accuracy of the model across different methods and select the best-performing one.

The report will provide insights into the performance differences between positive and negative reviews, evaluate the effectiveness of various deep learning algorithms for sentiment analysis, and present recommendations for building accurate and reliable sentiment analysis models. The findings of this project can be applied in various practical applications, such as customer feedback analysis, product or service improvement, and decision-making based on sentiment insights.

Dataset Description:

Sample Review from the dataset with a Label = 0(negative)

I grew up (b. 1965) watching and loving the Thunderbirds. All my mates at school watched. We played "Thunderbirds" before school, during lunch and after school. We all wanted to be Virgil or Scott. No one wanted to be Alan. Counting down from 5 became an art form. I took my children to see the movie hoping they would get a glimpse of what I loved as a child. How bitterly disappointing. The only high point was the snappy theme tune. Not that it could compare with the original score of the Thunderbirds. Thankfully early Saturday mornings one television channel still plays reruns of the series Gerry Anderson and his wife created. Jonatha Frakes should hand in his directors chair, his version was completely hopeless. A waste of film. Utter rubbish. A CGI remake may be acceptable but replacing marionettes with Homo sapiens subsp. sapiens was a huge error of judgment.

The entire dataset contains 40,000 movie reviews along with the sentiment label which is either 0 or 1.

Link: <https://www.kaggle.com/datasets/yasserh/imdb-movie-ratings-sentiment-analysis>

Project Description:

The goal of this project is to develop a sentiment analysis model using deep learning algorithms to analyze and classify feedback/reviews from audiences as positive or negative based on the sentiment expressed in the text. To prepare the raw data for analysis using machine learning techniques, we perform data preprocessing operations to enhance the dataset and make it computationally understandable.

1. Importing Libraries: Importing the necessary libraries to perform data analysis and natural language processing such as Pandas, Matplotlib, scikit-learn, and Natural Language Toolkit (NLTK).
2. Reading Data: Reading a movie review dataset using the read\_csv method of Pandas.
3. Data Cleaning: Checking if the dataset contains any missing values, and lower-casing and removing punctuations and stopwords from the text data using the Pandas and NLTK libraries.
4. Data Preparation: Splitting the dataset into training and testing data using scikit-learn, then converting the text data into numeric vectors using either CountVectorizer or TF-IDF Vectorizer from the scikit-learn library.
5. Model Building and Evaluation: Using a neural network-based approach, Tokenizer from Keras, to convert the text data into sequences, then padding them to make them of equal length before building and training a neural network model to classify the movie reviews. Finally, the trained model is evaluated on the testing dataset.
6. Data Visualization: Plotting a bar chart to visualize the number of positive and negative reviews in the dataset.

The data preprocessing operations include data quality assessment, data cleaning, data transformation, and data reduction. These operations ensure that the dataset is free from null values, inconsistencies, and unnecessary information, and is in a suitable format for further analysis. Specifically, the following operations were performed to preprocess the data:

1. Searching for and removing null values: Null or missing values in the dataset are identified and removed to ensure the integrity and reliability of the data.
2. Converting all values to lowercase: All text data in the dataset is converted to lowercase to standardize the text and prevent inconsistencies in case sensitivity during analysis.
3. Removing punctuation from the data: Punctuation marks, such as commas, quotation marks, and other special characters, are removed from the text data to focus on the meaningful words and reduce noise in the analysis.
4. Identifying and removing stop words: Stop words, which are common words like "they", "that", and "we'd" that do not carry much meaning in sentiment analysis, are identified and removed from the text data to reduce noise and improve the accuracy of the sentiment classification.
5. Encoding labels: The labels indicating positive or negative sentiment in the dataset are converted to numeric form, such as 0 for negative and 1 for positive, to enable the model to process and classify the sentiment accurately.

After preprocessing the data, we split the dataset into train and test data. The dataset is divided such that 20% of the data is allocated for testing, while the remaining 80% is used for training the model. This allows us to evaluate the performance of the model on unseen data and ensure its generalizability.

Classifications:

In the report, three different algorithms, LSTM, CNN, and RNN, were implemented and evaluated for sentiment analysis on text data.

The LSTM model used an embedding layer to convert words into vectors, processed them using an LSTM layer, and mapped the output to a binary output with a sigmoid activation function.

The CNN model had a architecture consisting of dense layers, an embedding layer, a convolutional layer, a max pooling layer, and an output layer with a sigmoid activation function.

The RNN model used recurrent layers for capturing sequential information.

Main reference papers used:

1. <https://www.researchgate.net/publication/328488167_Sentimental_analysis_using_recurrent_neural_network>

The report discusses the use of Long Short-Term Memory (LSTM) networks, which are a type of recurrent neural network, for sentiment analysis in Malayalam language. The report describes the implementation of a sentimental analyzer using neural networks and LSTM for Malayalam datasets.

The algorithm used for the implementation is outlined, including steps such as reading sentences and their tagged sentiments from a file, preprocessing the text by removing punctuation and splitting sentences into words, building a dictionary of words and mapping them to integers, converting sentences into arrays of integers, and splitting the data into training and test datasets. The LSTM graph is built with specific parameters such as the number of units in hidden layers, batch size, and learning rate.

The report also mentions the use of an embedding layer for word representation and the addition of dropouts to the inputs/outputs of the LSTM cells. The training data is converted into batches and fed into the model for training, and the test data is also processed in batches to calculate the accuracy of the model.

1. <https://cs224d.stanford.edu/reports/PouransariHadi.pdf>

The results of binary classification using different methods and classifiers were reported in this study. The baseline accuracy on the test data set was evaluated using the Python package scikit-learn. The following results were obtained:

1. Bag of Words + Random Forest Classifier: The accuracy obtained was 0.84352.
2. Bag of Words + SVM/Logistic regression: Elastic-net regularization was tried with different values of the elastic-net ratio α, and it was found that plain L2 regularization (α = 0) was optimal. Hyper-parameter tuning was then performed on the regularization parameter λ. The accuracy obtained for development and training sets are shown in the figure. The accuracy of the best model using SVM/Logistic regression was 86.6%.
3. Word2Vec + Averaging + Random Forest Classifier: Hyper-parameter tuning was performed on the context-size and word-vector dimension using the skip-gram model. It was found that larger context-size and word-vector dimension of 100 resulted in higher accuracy. The final test accuracy using the random forest, SVM, and logistic regression classifiers were 84.0%, 85.8%, and 86.6% respectively.
4. Word2Vec + Clustering + Random Forest Classifier: The accuracy obtained was 0.83528, which was not an improvement compared to the averaging method.

Additionally, recursive neural networks (RNTN) were used for sentiment analysis. The optimization process for standard RNTN was found to be expensive and slow, while a low-rank RNTN with rank 5 converged relatively faster. Thorough hyper-parameter tuning on learning rate, regularization strength, and rank was performed to obtain the best accuracy. The performance of the low-rank RNTN was compared to that of the standard RNTN.

Differences in Approach/Method:

w.r.t paper- 1:

* In our implementation, tokenization, padding, and vectorization are performed using the Keras library. The text data is first tokenized into individual words using the Tokenizer() function, and then the sequences of words are padded to have the same length using pad\_sequences() function. Finally, the padded sequences are vectorized using the Keras built-in function text\_to\_sequence().

In paper implementation, tokenization, padding, and vectorization are also performed, but using different libraries and methods. The text data is tokenized and vectorized using the CountVectorizer and TfidfVectorizer from scikit-learn library, and the sequences are padded using the pad\_sequences() function from the Keras library. The approach used in the second implementation is more flexible as it allows using different vectorization techniques based on the dataset and problem at hand.

* The paper implementation includes additional steps in building the LSTM graph, such as adding an embedding layer, creating multiple LSTM layers with a multi RNN cell, and adding dropout layers. Our implementation also specifies the number of units in hidden layers, batch size, and learning rate, but does not include these details uses a simpler approach to tokenization and vectorization.

Difference in Accuracy/performance:

* Results from our implementation presents a higher accuracy of 86.46% compared to the 80% achieved by the reference paper. However, it is difficult to compare them directly since they are based on different datasets, models, and evaluation methods. Both results suggest that deep learning techniques, particularly RNN-LSTM, can be effective for sentimental analysis tasks, but more work is needed to improve their performance on larger datasets.

Analysis:

Based on the accuracy results obtained, the LSTM model appears to outperform the CNN and RNN models in the task of sentiment analysis on the given text data. This could be attributed to the fact that LSTM and CNN models are capable of capturing complex patterns and correlations within the data compared to the simpler RNN model. Further analysis and experimentation can be conducted to fine-tune the models and optimize their performance for the specific sentiment analysis task.

Top of Form

Bottom of Form