NLP Sentiment Analysis for Phrases / Reviews

Jeyamaruthi Jayakumar[1], **Ankit Khanna**[1], **Rahul Tevatia**[1]
[jeyamaruthi.jayakumar, rahul.tevatia, ankit.khanna]@mavs.uta.edu

[1]The University of Texas at Arlington
Arlington, Texas, 76019

Abstract

Nowadays, before doing anything, we start by analyzing the internet, whether to do it or not. For example, before we go to a movie, we check the reviews for each movie. Before purchasing any product, we do the same, or before taking an online course, we read the comments. There are many other circumstances we do this. In this project, we are implementing different classification models to predict the sentiment of movie reviews, either as positive or negative, using only the text each review contains. We also introduce "JAR-20", a neural network model built from scratch to process the sentiment analysis.

1 Introduction

Sentiment analysis is the process of identifying whether a text generated or written by the user expresses a positive, or negative opinion/view. We start by gathering the dataset and cleansing it. Suppose the dataset collected consists of HTML tags or any other residual. We begin by cleansing to an improvised quality & we follow other methodology to pre-process the dataset. The cleansed dataset will be distributed into a ratio of 7:2:1 for training, test sets & cross-validation, then it is passed to the model.

We aim to provide step-by-step details on the process of analyzing sentiments using machine learning models and Neural networks on a large movie review dataset. Our main model for this task of sentiment analysis is the LSTM Sequential Neural Network Model. After we train and test our model, we are comparing our model with some other different machine learning algorithms for classification to find out the best model for sentiment analysis on the movie review dataset.

2 Dataset Description

The dataset we are using for our project is a Large movie review dataset. This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. It is a set of 25,000 highly polar movie reviews for training and 25,000 for testing. There is additional unlabeled data for use as well.

2.1 Training Dataset

There are a total of 25,000 training datasets, which are further classified into positive/negative with a count of 12,500 each. Each dataset is a .txt file that has a negative/positive review in the respective pos/neg file. The text files also have the rating given by the users.

2.2 Testing Dataset

It is similar to Training Dataset: same proportions & same count.

2.3 Data Preparation and Pre-processing

The main objective of data preparation and pre-processing of the reviews is to clean the noise (here words), which is not much relevant in finding the sentiment of reviews for example punctuations, special characters, numbers, and words or phrases which are less important to the context of the text.

Data preparation and processing in our project has been divided into some important phrases like:

Review and understand the provided Dataset
In this phase, we aimed to review and understand the provided movie dataset. And to figure what are the key issues we have to address for cleanup. Also, we will be cleansing the data.

Loading the Dataset
We then start working on all the files provided in both the pos and neg directories of the dataset. We load each file using file handling. We list each file in both directories and then load each file in turn.

Data Cleaning
Now we have our data loaded. We are going to preprocess the data so that we can pass that data to our model for training and can make better predictions from the model at the time of testing.
Initially, we are splitting the data into tokens and then using the Bag of Words strategy to build a vocab list i.e., we are tokenizing & we process these tokens for cleaning the data. The simplest way to tokenize a sentence is to use white space within a string as the delimiter. In python, we achieve this using the standard library method, split().
Some of the Data Cleaning tasks we are focusing on are:

- Lowercasing all text.
- Removing the Punctuation.
- Removing tokens that contain numbers.
- Removing stop words.
- Lemmatization.
- Removing tokens that don't have much importance.

After we perform all those cleaning tasks, we have a nice set bag of words or tokens from the first pass of preprocessing.

Building the Vocabulary
Doing the above data cleaning process, we generate a file named 'Vocab.txt'. The file consists of the bag of words.

Load Vocabulary or save processed data
Once we are done, with our preprocessing, cleaning, and vocabulary preparation phases, we use the vocab file with each file in positive & Negative files. To generate the preprocessed dataset named "positive.txt" & "negative.txt". This phase is all about saving the tokens to a vocabulary file.

3 Project Description

As we finish the Data preparation and Pre-processing steps, the main important task that we are focusing on is to build a model that classifies a review with a better accuracy score. Before we move towards the implementation of the model, we worked on creating features and labels and creating test and train data for the model.

3.1 Description

We introduced a sequential neural network model named "JAR-20", to do the sentiment analysis. Later on, we run the preprocessed dataset into the other well-known algorithms such as: Logistic Regression, Bernoulli Naive Bayes, Gradient Boost Classifier, Bagged Decision Tree Classifier & Support Vector Machine. We compared our model with other models and realized our model was the best among others. We will look in detail at what our model is composed of.

3.1.1 Our Model "JAR-20"

Layers: "JAR-20" has 3 LSTM layers: Each LSTM

layer has a unit of 16, 8, and 4 respectively. We show the hidden state of each input step by passing the parameter "return sequence" as True. Along with it, we have activation of "Sigmoid". Hence the classification will be of range 0 to 1. We add an optimizer name Adam to the model, to update the network weights iteratively based on the training dataset.

3.1.2 Other Models

The following algorithms are used for classification of the input dataset into the negative or positive case, these algorithms are also used to compare with our model by calculating the accuracy metric.

- Logistic Regression
- Bernoulli Naïve Bayes
- Gradient Boost Classifier
- Bagged Decision Tree Classifier
- Support Vector Machine

3.2 Main references description

We got the idea of how to preprocess the data i.e. we understood: why we need to remove HTML tags, stop words, and unnecessary words from the Learning Word Vectors for Sentiment Analysis [1]. We also attained the knowledge of lemmatization & Stemming from the same paper. Finally, while going through the IMDB Sentiment Analysis [4]. We understood that comparing the result of the existing work with our newly created "JAR-20" model will manifest the best resemblance of our model.

3.3 Approach

3.3.1 Overview

Our task of building a new approach for the sentimental Analysis was challenging, due to the gigantic dataset., but this challenge helps us to figure out the usage of a neural network by converting the dataset into NumPy array will be more efficient.
Specifically, the process is of **3 steps** i.e., **Preprocessing the data, converting the preprocessed data into a NumPy array,** and **classifying the result.**

For preprocessing you can refer 2.3.

Converting the preprocess data to NumPy array:
Once the tokenization is done, we can use the tokenized list to convert it into a **NumPy** array**.** We need to calculate, the mean, max & sum value before loading the data into our mode.
This will be helped to give max length while loading the data. The mean will help us figure out the maximum token value.
Finally, we inverse the keys and values of the matrix in order to fetch the text dataset.

Classifying the result:
Upon converting into the desired entity, we fit the dataset into our model with the validation split of 0.05 & epoch of 5 along with batch size → 64.
More we train the model; we saw the better the results were. Upon reaching maximum clarity the model stopped improving. At the same time, we also don't want the model to overfit.
The classification works as:
If the result of the model comes in the range of 0.0 – 0.49:
Then the movie review will be considered as Negative.
If the result is in the range of 0.5 – 1.0:
Then the movie review is classified as Positive.

After classifying we get the accuracy metrics & present our model how it works with help of some examples.

The comparison of "JAR-20" will be analyzed in section 4

3.4 Performance Measure

The best model from the referred paper is SVM, which gives an accuracy measure of 87.15%. Whereas the JAR-20 model gives a better prediction result for classifying positive/negative reviews with an accuracy of 93.33%.

3.5 Contribution in the Project

We introduced a sequential neural network model with 3 LSTM layer, to classify the movie review as negative/ positive.
Apart from that with the help of a reference paper & the machine learning course, we were able to build 5 existing models to compare with the "JAR-20". The 5 models are:

- Logistic Regression.
- Bernoulli Naïve Bayes
- Gradient Boost Classifier
- Bagged Decision Tree Classifier
- Support Vector Machine

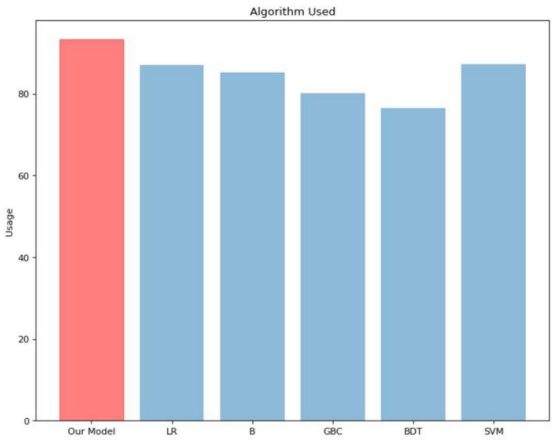
4 Analysis

A section of code exists to fetch an input (a movie review) from the user, which is loaded into the model to help us classify whether the review is positive or negative.

The accuracy of our model is: 93.33%, while compared to other models:

SNo.	Model	Accuracy (%)
1.	JAR-20 (Our Model)	93.33
2.	Logistic Regression	86.04
3.	Bernoulli Naïve Bayes	85.18
4.	Gradient Boost Classifier	80.18
5.	Bagged Decision Tree Classifier	77.07
6.	Support Vector Machine	87.16

Graph comparing our results:



4.1 What did we do well?

We utilized the neural network concepts to build our model and obtained better accuracy as compared to the other machine learning algorithms.

4.2 What could we have done better?

A user-friendly GUI could have been prepared for a more interactive experience with the model classifiers.

4.3 Future work

We can improve on our implemented models by introducing more pre-processing text feature extraction techniques like TF-IDF, Word2Vec, GloVe as one of the major disadvantages of BOW is that it discards word order thereby ignoring the context of words in the document.
Additionally, more sentimental analysis can be done to further classify the reviews into Partially Negative, Neutral, and Partially Positive. This can be implemented by using the rating value available in each folder of the notepad.

5 Conclusion

We have successfully developed a new model, which helps us to classify the movie reviews. We also developed 5 existing models to compare the results.
Finally, we were able to conclude that "JAR-20" has the best result compared to the existing models.

Acknowledgments

We are sincerely thankful to Prof. Jesus Antonio Gonzalez Bernal, Department of Computer Science & Engineering, The University of Texas at Arlington, for assistance in the project and for giving us an opportunity to implement the project.
Apart from this we thank Prof. Gonzalez for teaching us the Machine learning concepts.

References

[1] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts: Learning Word Vectors for Sentiment Analysis. (Links to an external site.)

[2] Andrew L. Maas, Andrew Y. Ng, and Christopher Potts: Multi-Dimensional Sentiment Analysis with Learned Representations. (Links to an external site.)

[3] Jean Y. Wu, Yuanyuan Pao: Predicting Sentiment from Rotten Tomatoes Movie Reviews. (Links to an external site.)

[4] Beatrice Lopez, Minh Anh Nguyen, Xavier Sumba: IMDb Sentiment Analysis.