A Progress Report

on

SENTIMENT ANALYSIS OF AMAZON PRODUCT REVIEWS

carried out as part of the course CSE CS3270 Submitted by

Naveesha Srivastava

209301372

VI-CSE

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

Computer Science & Engineering

MANIPAL UNIVERSITY
JAIPUR

Department of Computer Science & Engineering,
School of Computer Science and Engineering,
Manipal University Jaipur,
JAN-MAY 2023

Acknowledgement

This project would not have completed without the help, support, comments, advice, cooperation and coordination of various people. However, it is impossible to thank everyone individually; I am hereby making a humble effort to thank some of them.

I acknowledge and express my deepest sense of gratitude of my internal supervisor
Ms. Vibha Jain for her constant support, guidance, and continuous engagement. I highly appreciate his technical comments, suggestions, and criticism during the progress of this project "Sentiment analysis of Amazon product reviews".

I owe my profound gratitude to **Prof. Neha Chaudhary** , Head, Department of CSE, for her valuable guidance and facilitating me during my work. I am also very grateful to all the faculty members and staff for their precious support and cooperation during the development of this project.

Finally, I extend my heartfelt appreciation to my classmates for their help and encouragement.

Registration No. : 209301372

Student Name : Naveesha Srivastava

MANIPAL UNIVERSITY
JAIPUR
(University under Section 2(f) of the UGC Act)

Department of Computer Science and Engineering

School of Computer Science and Engineering

Date: 20/04/2023

CERTIFICATE

This is to certify that the project entitled "***Sentiment analysis of Amazon product reviews***" is a bonafide work carried out as ***Minor Project (Course Code: CS3270)*** in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering, under my guidance by ***Naveesha Srivastava*** bearing registration number ***209301372***, during the academic semester *VI of year 2022-23.*

Place: Manipal University Jaipur, Jaipur

Signature of the project guide:

Name of the project guide: Ms. Vibha Jain

Table of Contents

1. Introduction

Sentiment analysis is a natural language processing (NLP) method which is utilized to predict whether data is positive, negative, or neutral i.e., the sentiment of the data. Sentiment analysis is often performed on textual data.

A sentiment is an attitude, belief, or conclusion brought on by a sensation. It is commonly referred to as opinion mining, examining how individuals feel about particular things. The internet is a great resource for sentimental knowledge. Via various social media, including forums, blogs, and online social networking sites, online shopping wesbites, public can publish their own material and can openly lay their reviews.

In this project, we particularly look at sentiment analysis of an Amazon product based on the customer reviews. Each review corresponds to a rating from 1 to 5. The goal of sentiment analysis is to determine the overall emotional tone of a review.

Ratings and comments can be viewed as follows:

Rating	General meaning
1	Hated the product
2	Didn't like the product
3	No issue with the product, could expect more.
4	Liked the product
5	Loved the product

Most people today look at the product review and other customers' experiences with the product before buying it. This makes the product rating and reviews an important part of online shopping. According to recent statistics, 77% of consumers read product reviews before buying on Amazon.

Regardless of the approach used, sentiment analysis is a challenging task because of the complexity and variability of human language. There are a certain set of words or phrases through which a normal human being can easily identify the polarity of the review, but it becomes a lot more difficult for a computer to analyze a huge set of words, most of which are not necessary. Hence, data cleaning, pre-processing and other text analysis techniques come into picture, where tools extract useful information from it, which not only reduces the computational task, but also gives a fair accuracy about the polarity of the text.

Major use of sentiment analysis is for businesses to gain knowledge of its products through customer reviews. It provides crucial insights into customer preferences, which can be used to make better business decisions and improve customer satisfaction. For instance, organizations use sentiment analysis to identify popular products, track customer satisfaction over time, and monitor the reputation of a brand. It can help the company to gain insights regarding the scope of improvement in their products and services and start working on them before the product collapses from the market, and to make a better version of it. Sentiment analysis can be used to forecast sales based on the polarity of customer reviews. For instance, the scope of a new product can be predicted based on the sentiment of customer reviews.

1.2 Motivation

In today's world, everything is on our fingertips, just one click and people order thousands of items through online shopping. One such online commerce company is Amazon. But most often, people get confused regarding the quality of product they wish to order. Hence, it added a feature called as customer review, where the customer buys the product and put in a review for other buyers to get the knowledge of the product. We all know ratings and reviews are of utmost importance during online shopping. While they have only been around for about two decades, it is hard to imagine shopping without them.

Sentiment analysis is remarkably interesting in this field of studying the nature of the reviews. Customer reviews directly can't be processed by the computer. Hence, there is a need to pre-process the data to extract only the useful insights from the review. Depending on the review, the model trains to segregate it as positive or negative.

There are hundreds and thousands of reviews on a product and it becomes impossible for any company to go through these reviews line by line. Hence, the use of sentiment analysis becomes inevitable. This model helps to analyze thousands of reviews and show statistics about the product satisfaction of the customers, and hence, enhance the scope of development.

2. Literature review

In this section, we will only review a small portion of the prior literature. Roobaea Alroobaea[1] worked on Sentiment Analysis on Amazon Product Reviews using the Recurrent Neural Network (RNN). The contributions in the study are 1) Arabic data set are used in this research because few research have done in this area. 2) Used RNN to predict sentiment. 3) Compared their proposed model to Long short-term memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN). The proposed system succeeded in obtaining an accuracy of 85%.

Based on reviews, Hu and Liu[2] compiled lists of words that were both positive and negative. 2006 terms are on the positive list, while 4783 words are on the negative list. Both categories also contain several often used misspelt words in social media posts. They used a machine learning model called the maximum entropy model (also known as logistic regression) to classify opinion words and phrases into different sentiment orientations. They also used a rule-based algorithm to aggregate the classified opinions and generate summary opinions.

Pang and Lee [3] used a machine learning model called Support Vector Machines (SVM) to classify sentences as either subjective or objective. Then, they used a graph-based algorithm called the minimum cuts algorithm to summarize the subjectivity information in the text and compute the sentiment orientation of the text. Specifically, on the movie review dataset, their approach achieved an accuracy of 90.4% for identifying subjective sentences and an accuracy of 88.1% for predicting the sentiment orientation of the text.

Xu et al. (2016) proposed a cached LSTM model to capture the overall semantic information in a long text. The memory in the model is divided into several groups with different forgetting rates. The intuition is to enable the memory groups with low forgetting rates to capture global semantic features and the ones with high forgetting rates to learn local semantic features.

3. Methodology and Framework

3.1 System Architecture

Flowchart:

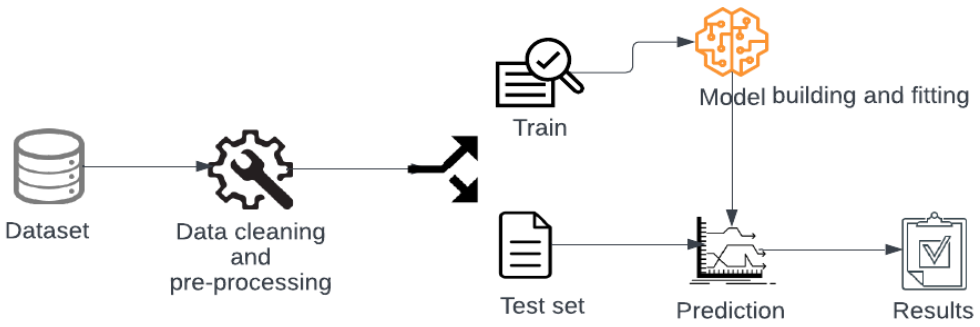

Figure 1 Flowchart for deep-learning pipeline

The basic steps in a binary sentiment analysis task are:

1. Preprocessing of the text data:
    - Tokenization of the review into separate words.
    - Removal of stop words, punctuation, and special characters, lemmatization.
2. Use of Word2Vec pretrained word embedding from TensorFlow hub.
3. Converting ratings from 1 to 5 into binary classification. Five rating as '1' and rest ratings as '0' using one hot encoding technique.
4. Training a binary classifier on the represented text samples(reviews) to predict the sentiment of new reviews using LSMT and RNN deep learning models.
5. Model testing.

3.2 Dataset

Amazon reviews for sentiment analysis - This dataset consists contains data about the customer's review details of an electronic product from Amazon. The major focus is on customer review and its corresponding rating for the sentiment analysis. Sample data is obtained from Kaggle.com

Number of tuples: 4915

Owner: tarık kaan koç

Link: https://www.kaggle.com/datasets/tarkkaanko/amazon

```
dataset.head()
```

	Unnamed: 0	reviewerName	overall	reviewText	reviewTime	day_diff	helpful_yes	helpful_no	total_vote	score_pos_neg_diff	score_average_rating
0	0	NaN	4.0	No issues .	2014-07-23	138	0	0	0	0	0.0
1	1	0mie	5.0	Purchased device , worked advertised . never m...	2013-10-25	409	0	0	0	0	0.0
2	2	1K3	4.0	works expected . sprung higher capacity . thin...	2012-12-23	715	0	0	0	0	0.0
3	3	1m2	5.0	think worked great.Had diff . bran 64gb card w...	2013-11-21	382	0	0	0	0	0.0
4	4	2&1/2Men	5.0	Bought Retail Packaging , arrived legit	2013-07-13	513	0	0	0	0	0.0

Figure 2 Dataset of Amazon product reviews for an electronic product

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4915 entries, 0 to 4914
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Unnamed: 0           4915 non-null   int64  
 1   reviewerName         4914 non-null   object 
 2   overall              4915 non-null   float64
 3   reviewText           4914 non-null   object 
 4   reviewTime           4915 non-null   object 
 5   day_diff             4915 non-null   int64  
 6   helpful_yes          4915 non-null   int64  
 7   helpful_no           4915 non-null   int64  
 8   total_vote           4915 non-null   int64  
 9   score_pos_neg_diff   4915 non-null   int64  
 10  score_average_rating 4915 non-null   float64
 11  wilson_lower_bound   4915 non-null   float64
dtypes: float64(3), int64(6), object(3)
memory usage: 460.9+ KB
```

Figure 3 Dataset information

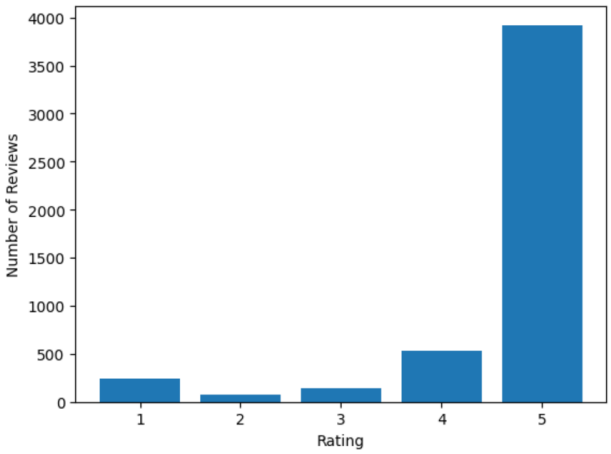

Figure 4 Frequency count of number of reviews per rating from 1 to 5

3.3 Algorithms and Techniques

Initial dataset contains the following reviews from the customers.

```
reviewText.head()

0                                          No issues.
1    Purchased this for my device, it worked as adv...
2    it works as expected. I should have sprung for...
3    This think has worked out great.Had a diff. br...
4    Bought it with Retail Packaging, arrived legit...
Name: reviewText, dtype: object
```

Figure 5 Initial reviews from the dataset

3.3.1 Data Cleaning

1. Tokenization

Tokenization is the process of dividing a piece of text into smaller units, called tokens. These can be words, phrases, or individual characters. In the context of sentiment analysis, tokenization is essential because it allows us to analyze the sentiment of individual words or phrases, hence making it useful to do predictions about the overall sentiment of the text. Additionally, tokenization makes it simple to eliminate stop words and punctuation from the text.

This project uses the "nltk.tokenize" module in the NLTK library which provides a collection of functions for tokenizing text data. The "word_tokenize" function is one of the most commonly used tokenization functions in NLTK. This function takes a string of text as input and returns a list of individual words or tokens, where each token is a separate word in the text.

reviewText = reviewText.apply(word_tokenize)

```
reviewText.head()

0                                    [No, issues, .]
1    [Purchased, this, for, my, device, ,, it, work...
2    [it, works, as, expected, ., I, should, have, ...
3    [This, think, has, worked, out, great.Had, a, ...
4    [Bought, it, with, Retail, Packaging, ,, arriv...
Name: reviewText, dtype: object
```

Figure 6 Reviews after tokenization

2. Removal of stop words

Stop words are commonly used words used in natural language but have little or no meaning in the context of text analysis. The removal of stop words from the text data is important in sentiment analysis because it can help to reduce noise and improve the accuracy and efficiency of the analysis.

By removing stop words, the model can focus on the words that are most meaningful and informative in determining the sentiment of a text. This project uses NLTK library to remove stop words. It provides a list of common stop words. Some of these are: ‘is’, ’it’, ‘are’, ‘the’, ‘an’, ‘and’, 'but', 'if'.

Removing negative words from the stop words list is a common practice in sentiment analysis. This is because negative words can significantly impact the sentiment of a sentence and removing them can alter the sentiment of the sentence altogether. Some other negative words include: “aren't", "couldn't", "didn't", "doesn't”.

reviewText=reviewText.apply(remove_english_stopwords_func)

```
reviewText.head()

0                                          No issues .
1    Purchased device , worked advertised . never m...
2    works expected . sprung higher capacity . thin...
3    think worked great.Had diff . bran 64gb card w...
4    Bought Retail Packaging , arrived legit , oran...
Name: reviewText, dtype: object
```

Figure 7 Reviews after removing stop words

3. Removal of punctuations

Removal of punctuations is important in text preprocessing because they do not carry any specific meaning and can interfere with downstream tasks like tokenization and classification. Punctuations can also create noise in the text data which can negatively impact the performance of a sentiment analysis model.

This project uses Python's ‘string’ module, which includes ‘punctuation’, a built-in string constant that contains a set of common punctuation characters. Some examples of punctuation marks included in the punctuation constant are commas, periods, question marks, exclamation points, and parentheses.

The code has an exception for removing punctuation because some punctuations may carry important meaning in sentiment analysis. For example, in sentiment analysis, the use of exclamation marks (!) can signify strong positive or negative emotions and removing them may alter the sentiment of the text. Similarly, removing periods (.) and commas (,) may affect the structure of the text.

reviewText = reviewText.apply(lambda x: ''.join([word for word in x if word not in punctuations]))

```
reviewText.head()

0                                          No issues.
1    Purchased this for my device, it worked as adv...
2    it works as expected. I should have sprung for...
3    This think has worked out great.Had a diff. br...
4    Bought it with Retail Packaging, arrived legit...
Name: reviewText, dtype: object
```

Figure 8 Reviews after removing punctuations

4. Lemmatization

Lemmatization is the act of breaking down words into their lemma, or root, form. In order to determine a word's underlying meaning, lemmatization involves reverting the word's inflected forms to their dictionary-original forms. For instance, "run" is the lemma for the words "running," "ran," and "runner." Lemmatization is necessary for sentiment analysis with the aim to increase analysis precision. By combining words with comparable meanings, it serves to reduce the dimensionality of the data, which leads to a more effective analysis procedure. We can guarantee that different spellings of the same word are treated as a single entity and do not skew the findings by breaking down words to their most basic form.

This project uses spaCy library for lemmatization. 'en_core_web_sm' is a pre-trained statistical model for the English language provided by spaCy. It contains a pipeline for various natural language processing tasks like part-of-speech tagging, dependency parsing, named entity recognition, and more.

```
lem = spacy.load('en_core_web_sm', disable=['parser', 'ner'])
```

In the above code, we are loading the en_core_web_sm model using spacy.load() function. The disable parameter is used to disable certain components of the pipeline to speed up processing and reduce memory usage. In this case, we are disabling the parser and named entity recognizer (ner) components as we do not need them for lemmatization. The parser in spaCy is used for syntactic analysis of the text whereas named entity recognizer (ner) in spaCy is used to identify entities like person names, organizations, and locations in the text. These are not necessary for lemmatization, so we can also disable tm to reduce computation time and memory usage.

```
reviewText= reviewText.apply(spacy_text_lemmatizer)
```

```
reviewText.head()
```

```
0                                              no issue .
1    purchase device , work advertise . never much ...
2    work expect . sprung high capacity . think mak...
3    think work great.had diff . bran 64 gb card go...
4    buy retail packaging , arrive legit , orange e...
Name: reviewText, dtype: object
```

Figure 9 Reviews after lemmatization

5. Word Embeddings

The word emeddings used in this project is a pretrained Word2Vec model called "Wiki-words-250" from TensorFlow Hub. This model is trained on a large corpus of Wikipedia text and generates 250-dimensional word embeddings. These are vector representations of words that capture their semantic meaning and relationships with other words in a text corpus.

```
encoded_reviews = get_word2vec_enc(reviews)
```

```
tf.Tensor(
[[-2.59923842e-02 -1.00169824e-02  3.52738537e-02 -3.18673439e-03
   2.78390143e-02  2.91083865e-02  2.30574328e-02  3.13552842e-02
  -1.47768389e-02  4.43636701e-02 -6.36036741e-03  2.22535096e-02
   3.64602096e-02 -5.08949831e-02  6.43709004e-02  7.32610375e-02
   4.10345979e-02 -9.39234793e-02  1.07663488e-02 -7.12433681e-02
```

Figure 10 250-Dimension Word2Vec embedding of the first review

Ouput is a sequence of word2vec embeddings, one for each word in the first review of the dataset. Each embedding is a NumPy array of floating-point values.

6. Padding

Padding is needed because most algorithms require that all inputs have the same shape. In the case of text data, this means that all sentences must have the same length. The function pads the encoded sentences with zeros so that they are all the same length (50 in this project). Padding is particularly important for RNNs because they process sequences of inputs one step at a time. This allows the RNN to process the inputs in parallel, which can speed up computation.

padded_encoded_reviews = get_padded_encoded_reviews(encoded_reviews)

7. One-hot encoding

The one-hot encoding is a technique used to convert categorical data into numerical data that is more easily consumed by algorithms. In this case, the ratings are categorical, either '5.0' or '1.0'. The one-hot encoding will represent these two categories as binary vectors, where each vector has a single 1 indicating the category it represents.

encoded_rating = [rating_encode(rate) for rate in rates]

The function rating_encode returns a list with two elements, either [1, 0]: for 5 or [0, 1]: for 1.

3.3.2 Model Implementation

Build model using RNN + LSTM

This model uses sequential model of tensorflow. A sequential model object is created and adds an LSTM layer with 32 units(dimension of the hidden state) to it. Then, a dense layer with 2 units(dimension of the output state) and a softmax activation function(it converts a vector of value to a probability distribution.) is added on top of the LSTM layer. The model is then compiled with categorical cross-entropy loss, the Adam optimizer, and accuracy as the evaluation metric. This trains the model using the training data (train_X and train_Y) for 25 epochs. An epoch is one iteration over the entire training dataset. During training, the model is updated to reduce the loss and improve its performance on the given task. The model.summary() function prints a summary of the model architecture.

In the context of natural language processing, the input to a neural network is usually a set of word embeddings. Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN), capable of learning long-term dependencies.

An LSTM layer consists of a set of memory cells that are connected by gates. The gates regulate the flow of information into and out of the memory cells, allowing the LSTM layer to selectively remember or forget information from the input sequence. The LSTM layer takes as input a sequence of word embeddings, and produces as output a hidden state vector that represents the information learned from the input sequence. In neural networks, the activation function is a function that is used for the transformation of the input values of neurons. Softmax activation function is used in this project that scales numbers/logits into probabilities. The output of a Softmax is a vector (say v) with probabilities of each possible outcome. The probabilities in vector v sums to one for all possible outcomes or classes.

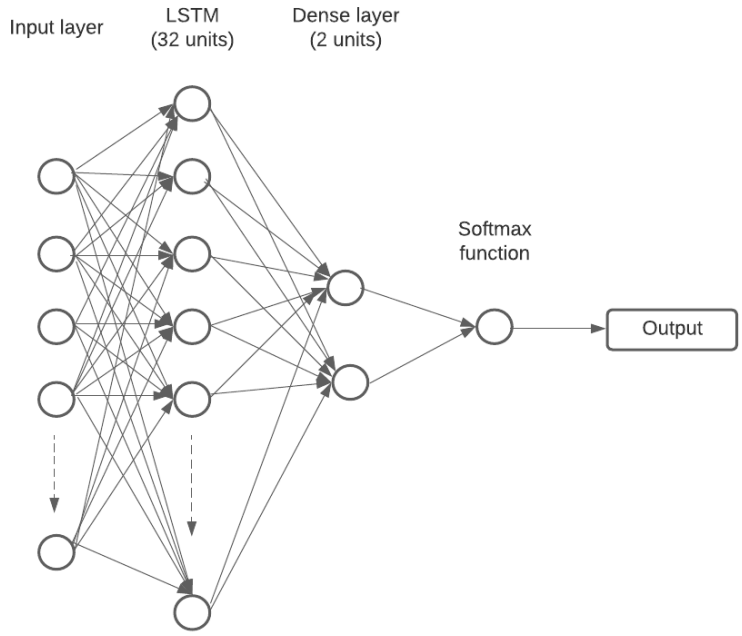

Figure 11Working of RNN-LSTM model

4. Work Done

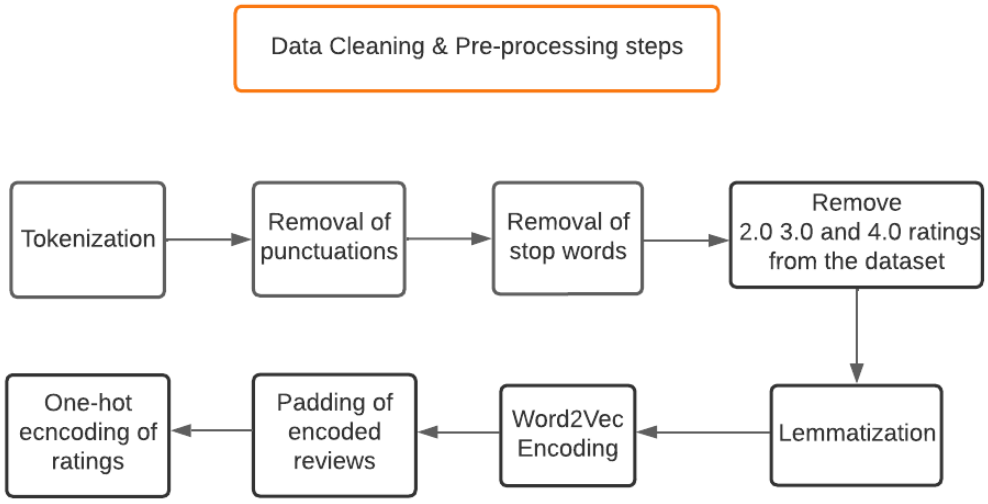

Figure 12 Steps involed in data pre-processing

Model training and testing

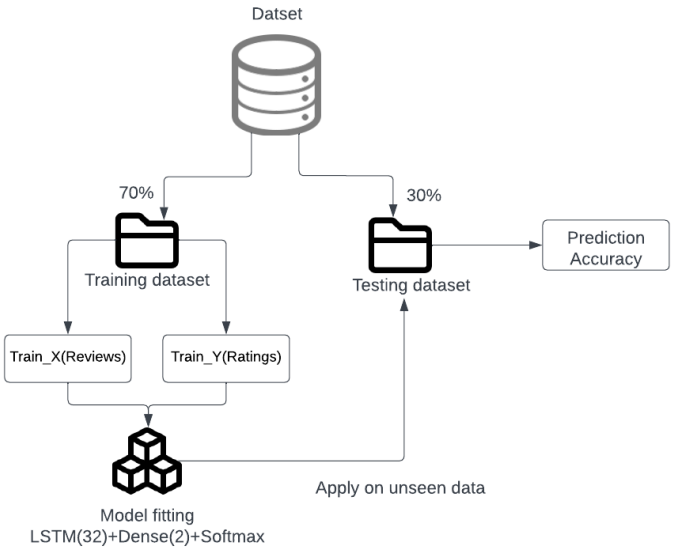

Figure 13 Overview of dataset splitting into 70:30 train-test ratio and model testing on the unseen dataset to predict the accuracy

5. Result and Discussion

Parameters and results:

1. Total data	4914
2. Split ratio	70:30 train-test ratio
3. Training data	3439
4. Testing data	1474
5. Model used	RNN- LSTM
6. LSTM layer units	32 units
7. Dense layer units	2 units
8. Activation function	Softmax Activation Function
9. Number of Epochs	25
10. Test Loss	0.1114
11. Test Accuracy	0.9653225541114807

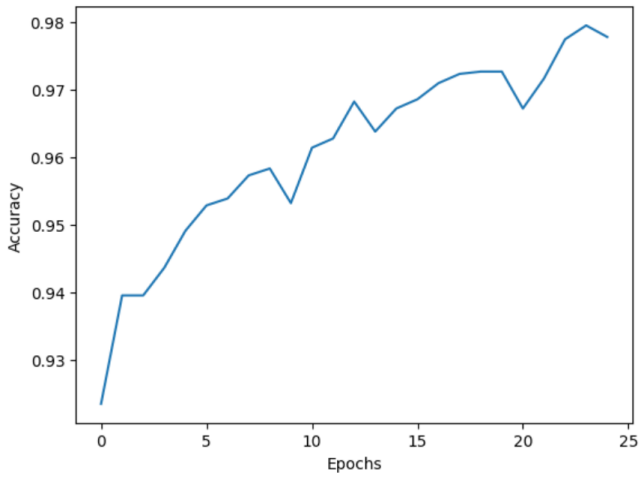

Figure 14 Accuracy score plotted against each epoch. The model reached its highest peak with the accuracy score of 0.9789

Other results:

Precision: [0.97478992 0.74]
Recall: [0.98891731 0.55223881]
F1 Score: [0.98180279 0.63247863]

Confusion Matrix:
[[1160 13]
[30 37]]

In the results, the precision, recall, and F1 score values are given in the form of 1-D arrays, where the first element corresponds to the "positive" class (in this case, a rating of 5) and the second element corresponds to the "negative" class (in this case, a rating of 1).

Precision is the ratio of true positive predictions to the total number of positive predictions. The precision for the positive class is 0.974, which means that out of all the reviews that the model predicted as positive, 97.4% of them were actually positive.

Recall is the ratio of true positive predictions to the total number of actual positive instances. Recall measures how many of the actual positive instances are correctly identified by the model. The recall for the negative class is 0.552, which means that out of all the actual negative reviews in the dataset, the model was able to correctly identify 55.2% of them.

F1 score is the harmonic mean of precision and recall. F1 score is a better metric than accuracy in cases where the classes are imbalanced, i.e., one class has significantly more samples than the other. The F1 score for the positive class is 0.98.

The confusion matrix shows the number of true positives, false positives, true negatives, and false negatives for each class. In this case, the confusion matrix shows that the model correctly identified 1160 positive reviews and 37 negative reviews, but misclassified 13 positive reviews as negative and 30 negative reviews as positive.

6. Conclusion and future plan

Sentiment analysis is remarkably interesting in this field of studying the nature of the reviews. Customer reviews directly can’t be processed by the computer. Hence, there is a need to pre-process the data to extract only the useful insights from the review. Depending on the review, the model trains to segregate it as positive or negative.

There is a scope to improve the dataset quality by removing the biasness in the data. Approximately 80% of the reviews are 5-star rated. This will give more accurate results on how efficient the model is. Analysis of emoticons is another challenge to be handled, because it has been observed that a lot of reviews contains emoticons, which directly imply the true sentiment of the text. Studying relationship between sequence of emoticons is also a challenging yet necessary task for better analysis. Multilingual text analysis is yet another challenge in Sentiment Analysis.

7. References

[1]Alroobaea, Roobaea. "Sentiment Analysis on Amazon Product Reviews using the Recurrent Neural Network (RNN)." International Journal of Advanced Computer Science and Applications 13.4 (2022).

[2] Hu M, Liu B (2004) Mining and summarizing customer reviews In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 168–177.. ACM, New York, NY, USA.

[3]Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts In: Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, ACL '04.. Association for Computational Linguistics, Stroudsburg, PA, USA.

[5] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[6] Wang, H., & Jiang, J. (2017). Recurrent neural network for sentiment analysis on twitter data. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 989-999).

[7] Tai, K., Socher, R., Manning, C. D., & Ng, A. Y. (2015). Improved semantic representations from tree-structured long short-term memory networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers) (pp. 667-672).

[8] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[9] Maas, A. L., 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-Volume 1 (pp. 142-150). Association for Computational Linguistics.

Websites:

1. Kaggle."Amazon Fine Food Reviews." Kaggle, n.d., www.kaggle.com/snap/amazon-fine-food-reviews.
2. "NLTK stop words", Python Tutorials, https://pythonspot.com/nltk-stop-words/
3. "Introduction to Long Short Term Memory (LSTM)", Analytics Vidhya, https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/
4. "word2vec", TensorFlow, https://www.tensorflow.org/tutorials/text/word2vec
5. A Complete Understanding of Dense Layers in Neural Networks By Yugesh Verma, https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/