Identifying fake users using an integrated system using NLP

MSc Research Project
Data Analytics

Karthik Ramachandran
Student ID: x21234884

School of Computing
National College of Ireland

Supervisor: Taimur Hafeez

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Karthik Ramachandran
Student ID:	x21234884
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Taimur Hafeez
Submission Due Date:	14/08/2023
Project Title:	Identifying fake users using an integrated system using NLP
Word Count:	7450
Page Count:	20

Signature:	KARTHIK RAMACHANDRAN
Date:	18th September 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Identifying fake users using an integrated system using NLP

Karthik Ramachandran
x21234884

Abstract

Fake reviews on online platforms have emerged as a alarming issue. Consumer perceptions on any product is influenced by deceptive or biased feedback posted in online platforms. The integrity of product or service ratings can be distorted by these fake reviews, which can also undermine user confidence in buying the product. This study proposed a novel approach on detecting users who post fake reviews with an integrated system using embedding vectors and clustering similar reviews using cosine distance. A Long Short-Term Memory (LSTM) model was built to predict the user ratings and this model is used to transform each review to its own embedding vectors. Grouping of similar embedding vectors is achieved by finding cosine angle between the vectors. Google's ScaNN was implemented to calculate the cosine distance between the vectors and for faster search and retrieval. The model was built with an accuracy of 78% for test data and the grouping of similar reviews were achieved.

1 Introduction

User-generated reviews have become increasingly important in influencing consumer decisions as a result of the wide-spread use of online platforms. But as online reviews have gained popularity, fake reviews have also appeared. These can deceive customers and jeopardize the trustworthiness of these platforms. To ensure authenticity and reliability in online consumer feedback, spotting and eliminating fake reviews has become crucial. The conventional methods for detecting fake reviews were through manual intervention. Later, variety of machine learning algorithms, such as supervised classification techniques, feature engineering, and sentiment analysis were used to flag the fake reviews.

This research addressed the research question: *How well can NLP implemented to flag negative comments in amazon platform and group these negative reviews posted from the same users using historical data.*

The temporal dependencies present in the reviews data made it difficult to classify the label using conventional methods. In order to eradicate this limitation, deep learning techniques like Long Short-Term Memory (LSTM) networks is used which are excellent in modeling sequential data. Recurrent neural networks (RNNs) of the LSTM have implemented in a variety of sequential data models with an exceptional performance including speech recognition, machine translation, and text generation. The distinctive architecture present in LSTM models helps to capture long-term dependencies and store lengthy data sequence. This peculiar property of LSTM networks can be examined and comprehend the temporal dynamics of reviews or comments which enhances to label fake reviews. This study aim to build a reliable and effective LSTM model that can label reviews between real and fake with high accuracy by taking advantage of the sequential type of review data. In addition, this study aims to improve the performance of the model by concentrating on important features of the review text which then suppress the noise and ignore irrelevant information. This research ensures the limitation of previous approaches is been addressed including clustering similar reveiws and adds value to the study. An improved accuracy and robustness in distinguishing between genuine and fake reviews is anticipated while building LSTM networks. As a result, the customer trust can be restored, foster fair competition among businesses, and preserve the integrity of online platforms.

The embedding layer in LSTM model is extracted for transforming input tokens to dense vectors. The semantic relationships and information from a context is retrieved from the textual data using the embedding layer. This transformation is vital for natural language processing models. In this study, these vectors were using to find the similarity between the reviews. The similarity between the vectors is calculated by finding the angle between the vectors. The cosine of the angle between these vectors, indicating how closely the vectors aligned in high dimensional space. This research use Scalable Nearest Neighbors framework to find nearest neighbor effectively. Scann (Scalable Nearest Neighbors) is an algorithmic framework created to efficiently and effectively carry out nearest neighbor search in high-dimensional spaces. The Scann algorithm leverages techniques such as projection of vectors and quantization with the ability of partition and indexes the high dimensional space, enabling faster and more efficient nearest neighbor search. The integrated solution of Scann with cosine similarity for calculating the distance can achieve scalable similarity search with high efficiency.

After the introductory segment, the subsequent section (Section 2) discussed the relevant literature, while Section 3 expounds upon the methodology presented in this research. Section 4 encapsulates the design, specifications, and a visual flowchart. The various implementation stages are outlined in Section 5. Section 6 describes the research's evaluation and metrics, while the concluding remarks and prospects for future investigations are captured in Section 7.

2 Related Work

Researchers have used multiple traditional algorithms including Random Forests, Logistic regression and Support Vector Machines (SVMs) to detect fraudulent reviews. These models were employed to identify the irregularities in user review like sentiment analysis, feature engineering and recognising linguistic patterns. Additionally, few researchers have also explored graph-based models to label credible reviews. Recurrent neural networks (RNNs) and Transformer-based models have demonstrated promising results by capturing contextual details and semantic dependencies in reviews. The expert opinion from below mentioned researchers is also taken into consideration while creating this system.

2.1 Detection of fake reviews using different models

The author Tang et al. (2021) proposed an ERNIE-BiLSTM model for the detection of fake commodity reviews. The review of this paper provides an overview of the implementation of model and highlight the significance and contributions of the paper. This paper is titled as ”A Study on Detection of Fake Commodity Reviews Based on ERNIE-BiLSTM Model” and was presented at the International Conference held in Kunming, China. Fake reviews have become a common issue on online platforms where customers are mislead which influence their purchasing decisions. Multiple studies have been conducted to develop techniques for detecting fake reviews, including both content-based and behavior-based approaches. Introduction of content based methods examine textual features, linguistic patterns, and sentiment analysis while behavior based methods consider reviewer characteristics, temporal patterns, and social network analysis. The author proposed a novel approach that combines ERNIE model, which incorporates external knowledge with Bidirectional Long Short-Term Memory (BiLSTM) model. The textual feature representation is improved by incorporating outside information in ERNIE model whereas contextual information is captured effectively in Bi-LSTM model. The proposed model is trained on the reviews which is mapped to a label - real or fake. The discriminative features were generated by the model which is used to predict the label precisely. The paper mainly discussed on the application of the ERNIE-BiLSTM model for labeling commodity reviews. This model outperformed the baseline models in terms of performance by leveraging the strengths of both ERNIE and BiLSTM model. The proposed model's efficacy is confirmed by the experimental findings reported in the paper. The paper concludes by presenting a novel method for identifying fake commodity reviews using the ERNIE-BiLSTM model. The combination of the trained language models with the deep learning model is used for fake review detection. The study results show the potential to implement in real world scenario to address the problem of fake online platform reviews.

The paper titled ”Detection of Fake Online Hotel Reviews” was presented at international conference for ICITST (Internet Technology and Secured Transactions) in Cambridge, UK by Sandifer et al. (2017). This study addressed the detection of fake reviews of hotel posted on online platforms. Online users face a challenge due to the prevalence of fake reviews, which can skew judgment and degrade hotel's reputation. The authors' goal is to create a technique for accurately identifying fake hotel reviews. In the paper, fake review detection is discussed using a content-based methodology. The reviews are transformed to tokens and most of the features are extracted including syntactic patterns and the presence of multiple occurrence of tokens. Multiple algorithms including Multinomial Naïve Bayes, Bernoulli Naive Bayes, and logistic regression models were implemented and the performance is evaluated. Multinomial Naive Bayes out performed comparing with other models and yielded an accuracy of 85.9The authors divided the dataset of hotel reviews that included both real and fake reviews into train and test to build and evaluate the model. They demonstrate the efficacy of their strategy through experiments where high accuracy was achieved identifying real from fake hotel reviews. To conclude, paper focused on detection of fake online reviews posted about hotel to degrade the reputation. The authors described their methodologies and findings with the labeled dataset and used part-of-speech features to analyze the reviews. Three different classification models were employed and result stated Multinomial Naive Bayes classification model built with higher accuracy than other models. The authors highlighted future work feature extraction methods, incorporating additional classification models, and introduction of voting algorithms. These steps aim to improve accuracy and built an effective detection model.

The research paper ”Spam Review Detection Using Deep Learning” was presented at Information Technology Electronics and Mobile Communication Conference in 2019 by Shahariar et al. (2019). The authors concentrated on the issue of spam review detection and suggested a deep learning-based solution to it. This study helps to create a quick and accurate method to label spam reviews, which will mislead many consumers during product purchase due to false information. Spam reviews can be detrimental to both consumers and businesses because they can influence decisions to buy and skew public perception of a good or service. The authors developed a deep learning model Long Short-Term Memory (LSTM), a recurrent neural network (RNN) to address the issue. The reason for choosing LSTM model is because of its potential of handling sequential data. The discovery of the patterns and traits connected to spam reviews is achieved by building LSTM model which was trained on a sizable dataset of reviews. The authors proposed two types of model including traditional and deep learning for spam review detection with four phases. Data collection and preprocessing, including the use of NLP methods, are part of the first phase. As the data holds both the labelled and unlabeled data, Active Learning Algorithm is used for labeling the unlabeled data which represented as second phase. n-grams, Word Embedding's, and TF-IDF are just a few of the feature selection methods that were focused of the third phase. In the final stage, classifiers for deep learning (MLP, CNN, LSTM) and traditional machine learning (SVM, KNN, NB) are both used to detect spam. The performance comparison demonstrates that deep learning classifiers perform better than conventional machine learning techniques. The authors conducted experiments in three different versions of dataset. The experimental results of CNN and LSTM over the "Ott Dataset" was marked as first experiment. The second experiment includes CNN & LSTM over the "Yelp Dataset". The third experiment showcased the findings of MLP model over Ott and Yelp datasets. The final experiment displayed the results of conventional classifiers, including K-Nearest Neighbor (KNN), Support Vector Classifier (SVC) and Naive Bayes (NB) over Yelp dataset. Authors concluded stating the limitations and the future works including enhancing the data labeling process by incorporating Deep Learning methods, introduction of pre-trained word representation vectors like GloVe to evaluate the performance, exploring hybrid CNN-RNN model.

In another study by Yao et al. (2019), a spam review detection model is built and was presented in IEEE 21st International Conference on High Performance Computing and Communications. The study suggests a technique for identifying fake reviews by comparing the format and style which appear on different review platforms. The paper discusses the increasing importance of user reviews on websites like Yelp, IMDB, and Amazon. Multiple experiments were conducted by the researchers to demonstrate the efficacy of their detection strategy. These results were evaluated with historical studies. The objective is to identify the frequency of review appearances and its reliability. The implementation is involved with multiple steps. The first step involves gathering reviews of target product from target site. Later, reviews were collected from different comparison site. Subsequently, abnormal periods, which are possibly linked to spamming activities, are identified by comparing the rate of change in the number of reviews and the rate of change in the review ratings across the target site and the comparison sites. Finally, a classification task is implemented to label the reviews between normal and spam during abnormal periods. This classification was achieved by validating the harmony between review rating and the review text with sentiment analysis. The abnormal periods were identified with 3 criteria's including - The count of reviews in target site should be increased, the growth rate of target site must be higher than other comparing sites and the average rating of target site should be greater than the comparison sites. The above criteria should be met during the posting period. The experiment results were showcased with more than 80% classification accuracy for two target datasets including imdb and fandango. For imdb, the precision value for detecting spam review was reported at 58% which is less comparing with Fandango with 75%. To conclude, this research paper suggested a novel approach on detecting spam reviews from multiple reviews sites than focusing on single site. The results obtained from the experiments indicated the proposed method outperforms existing approaches in detection of spam reviews during abnormal periods.

The study conducted by Ibrahim et al. (2017) on spam detection was submitted at 2017 IEEE Conference on Application, Information and Network Security (AINS). This study focused to improve review spam detection and classification using ensemble classifiers which is integrated using three classifiers including logistic regression, support vector machines and naive Bayes and an Arching classifier. The authors mentioned the challenges in identifying and preventing spam reviews in existing online portals. Researchers also compared the results of different historical models including supervised, unsupervised and also ensemble methods. However, authors still identified the gaps in effectively classifying the spam reviews. Researchers have undergone several pre-processing steps to produce high quality data for modeling. This includes elimination of stop words which is essential for classifier model and stemming to create the base form of a vocabulary. Authors claimed the result of pre-processing reduced the corpus size and later corpus is converted to vectors. Multiple packages were used for dimensional reduction and feature extraction. Authors used WEKA application for pre-processing, labeling the reviews and for creation of ensemble model. The technique of ensemble classifier is invoked when the output of the model is different for certain inputs. In the case of disagreement of output, which is known as ensemble diversity, author proposed the voting mechanism for the classification. Authors also proposed other ensemble diversity techniques and the best technique is chosen in terms of accuracy. In this study, models were trained on fraction of dataset and the final label is taken by averaging the combination of results. The bagging of each sample is different which helps to provide a unique approach to the problem. The precision and recall for the ensemble classifiers shows higher value while comparing with individual classifier. The test data metrics is evaluated using F1- score which take both spam precision and recall into consideration. In comparison with the individual classifier model, ensemble models classify the reviews more accurately. Authors also reported the metrics of individual classifier where logistic regression predict less accurate. This is followed by Support vector machine and Naive Bayes. The ensemble classifier outperformed with 8.1% higher than Naive bayes.

Unlike other studies identifying individual spammers, the study conducted by Cao et al. (2021) and his team helps to identify fake reviews posted by collective people or group. In the paper, a fake reviewer group detection technique called REAL is implemented by using modularity based graph clustering method. To achieve graph clustering authors applied spectral modularity and Graph Neural Networks (GNNs) to enhance structural information at higher degree which help to identify fake individuals group. To detect anomalies at individual and group level, the REAL methodology consider three main factors including patterns, relationship between the vocabularies and user behavior. The proposed method improved the accuracy and precision in identifying large groups after implementing in real dataset. Author validated the effectiveness of REAL technique. The study examined the impact of cluster size in the datasets including YelpNYC and YelpZip. The mean precision value of top 10 clustered is noted which is relevant for next detection stage. The results demonstrate that cluster precision increases when the number of clusters is relatively small, as the smaller group sizes theoretically improve precision. In contrast, cluster precision falls off as the cluster number increases. Author mentioned the reason for this drop could be the availability of too many cluster which lead to fewer groups being detected and precision is negatively impacted by smaller group sizes. The study compared REAL with other baselines like ColluEagle, DeFrauder, and GroupStrainer and found that ColluEagle achieves high precision. The problem with ColluEagle is that they were unable to extract groups of large sizes, typically detecting groups with only 2 to 4 people. As fake reviewer groups frequently consist of more than a few people, the authors question the efficacy of such results. Finally, REAL outperforms the majority of baselines in the experimental evaluation, striking a satisfying balance between group size and precision.

In another study by Thahira and Sabitha (2021) concentrated on detecting spammer groups by proposing a framework using graph algorithm. In addition to introducing a feature called group rating similarity (GRS), which is based on the score rating given for each reviews, they also considered five features which used for group spamming. Any two reviewers who frequently discuss the same target products are given an edge and the reviewer set becomes the vertex set. The edge's weight indicates the degree of collusion between the two reviewers. The edge weight is determined using two factors including review posting date and the rating score. Authors also added that spammers can't manipulate these factors and play a vital role in identifying spam groups. Cumulative distribution function is used to validate the effectiveness of group spamming features. Authors concluded stating new features added to framework have enhanced to get better precision. Introduction of GRS feature helped to derive high discrimination capability and only drawback witnessed was the high execution time while comparing with existing system.

Another research paper fromLiu et al. (2017) suggested a behavior-based approach to identify spam in reviews. Two algorithms were introduced to find similar and pertinent reviews across all reviews. The review identification algorithm executes faster than conventional ones. In addition, implementation of automatic word segmentation examines the relationship between the review content and the specified topics. The experiment was divided into three sections: 1) A similarity experiment comparing the performance of conventional and their own similarity algorithms on five sets of data, with their algorithm demonstrating better performance; 2) A relevancy experiment using their own relevancy algorithm to classify reviews based on relevancy ranks; and 3) Human evaluation, where three humans independently verified the experimental results of 1000 mobile phone reviews.The authors concluded stating similarity algorithm outperformed the conventional one; and algorithm performed well on a large dataset of 214,005 reviews, categorizing reviews with a similarity of over 70

2.2 Cosine similarity related works

The author Srivamsi et al. (2023) proposed a system in biomedical field that gain insights from text data. She transformed all text data into word vectors using a vector search engine called Milvus. The numerical vectors derived from words of text is represented in a high dimensional space. The model selection is always a crucial part and this paper highlighted a BERT model called en_core_sci_scibert which was trained on biomedical literature. This model helps to create the word embeddings. Similarity between the vectors is assessed using cosine similarity, where the angle between the vectors determines the similarities between them. The author concluded stating the new proposed system was able to map the biological information text with a similarity value of 0.8. This approach helps to yield new insights from massive biological dataset.

In another research by Ramadhanti and Mariyah (2019) for identifying text duplication by Ramadhanti, TF-IDF method is employed on Indonesian Wikipedia articles for assigning weights on the basis of word frequency. Cosine similarity algorithm is executed on these word vectors. The two comparison method proposed by this study were simultaneous and partial comparison. In partial comparison, similarity calculation is performed after segmenting the documents but in Simultaneous similarity score is calculated directly on documents. More accurate similarity results were obtained from partial comparison technique. The TF-IDF method using Unicheck application was set as the benchmark for this study and the difference is noted due to disparity in methods. This study also proposed model using Word2Vec which helps in achieving better result due to the fact that TF-IDF fails in paraphrase detection.

The study conducted by Soyusiawaty and Zakaria (2018) on similarity Detector helps to identify alternative books if the actual book is out of stock. Author mentioned more than 80% people couldn't find any alternative similar books in a library. She proposed a book search engine that helps to locate books with content. Author create a web based application which extracts the summary and compare it with rest of the books. The comparison result showcased similar books with a ranking column. After implementation in testing phase, author claimed the system could able to get 82.14% relevant match rate, which demonstrates its applicability in helping users locate substitute reference books. Author further recommends to extend this study using different similarity search which can widen current scope of project.

The summary of above literature are provided in Table 1

Table 1: Summary of related works

Author	**Techniques**	**Results**
Tang et al. (2021)	ERNIE-BiLSTM	Accuracy 89%
Sandifer et al. (2017)	Multinomial Naïve Bayes	Accuracy 85%
Shahariar et al. (2019)	LSTM	Accuracy 96%
Yao et al. (2019)	Abnormal period detection	Accuracy 85%
Ibrahim et al. (2017)	Naïve Bayes, SVM and Logistic Regress.	Accuracy 89%
Cao et al. (2021)	Graph CNN, Anomaly indicators	Precision 0.82
Thahira and Sabitha (2021)	Group Rating Similarity	Precision 0.56
Liu et al. (2017)	identification indicators	Accuracy 46%

3 Methodology

There were different approach to implement a data driven system. The methodology Knowledge Discovery in Databases (KDD), is employed in the proposed study. Multiple phases of KDD for data mining is implemented in this research, to build the relationships, identifying the patterns and trends within the datasets. The phases of KDD are in sequential order which can be revisted in an iterative manner. The proposed methodology is shown in Figure 1

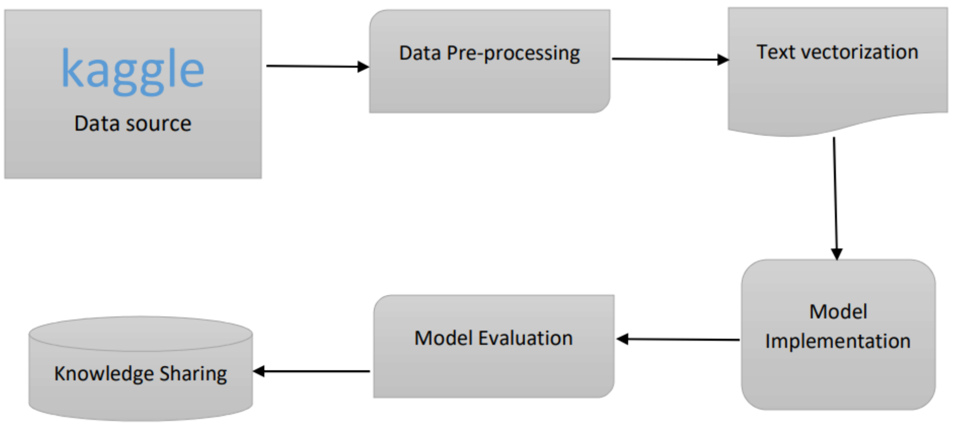


Figure 1: Methodology

3.1 Data Extraction from the datasource

- The initial step involves obtaining the necessary data, which is extracted from Kaggle. The dataset contains 21000 rows with 9 columns. The relevant columns used for this study include amazon reviews text, rating provided by user, and a label columns stating the reviews are fake or real. The dataset was provided in csv format with structure defined. The dataset contains information about products, user reviews, and corresponding ratings given by users for these products. [1]

3.2 Exploratory data analysis on Amazon dataset

As mentioned earlier, the two crucial columns utilized in this study are the user reviews and their corresponding ratings. The distribution of the target column, i.e., the Rating column, is depicted in Figure 2

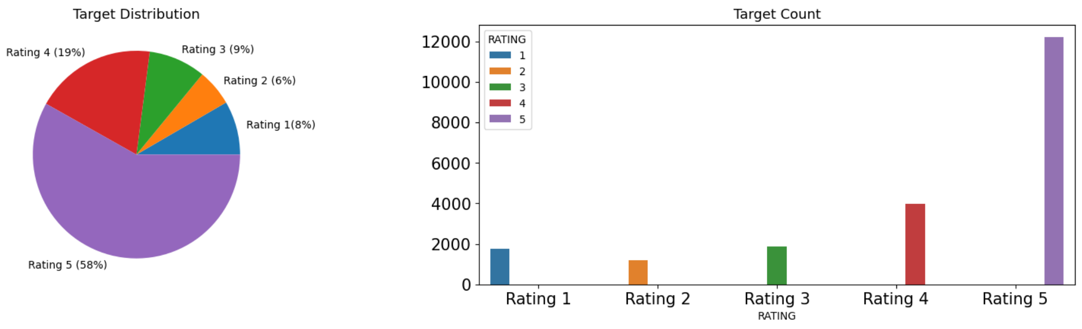


Figure 2: Distribution of rating

[1]Data URL: `https://www.kaggle.com/datasets/lievgarcia/amazon-reviews`

The text review column was examined, and it was found that there are no rows with a length of fewer than five words

3.3 Preprocessing on Amazon dataset

Constructing an LSTM-based natural language processing (NLP) model involves several crucial stages. The first step is data cleaning, which is vital before model development.

3.3.1 Excluding noisy data

Including irrelevant data in the input can significantly compromise the model's performance. For this LSTM model, the input consists of a free-text comment field, encompassing both recognized vocabulary and non-vocabulary items. In this study, the comment field contains various HTML tags, which were initially eliminated. Subsequently, tokenization and punctuation removal were accomplished using a text vectorization layer, which will be discussed in later sections. Furthermore, the data was further refined to exclude any non-text items present in the review columns, such as audio and video tags

3.3.2 Label encoder

The target variables in a NLP model need to pre-proceesed before sending to model.In this study, IntegerLookup layer is used to transform the target variable to one-hot numerical representation which is fed into network architecture. The dataset's ranking column contains 5 distinct values. The resulting output from this stage will comprise 6 distinct representations, with an extra one denoting "Unknown."

3.3.3 Text vectorization for review column

The text data from comments column is transformed into a numerical format using text vectorization layer tf.keras.layers.TextVectorization. The parameters used for this study includes-

- Max tokens- This parameter defines the value that maximum tokens can be accepted for text vectorization. Max token is assigned with value '100000'. Tokens outside the vocabulary is treated as 'OOV' token
- standardize- This parameter is set to True where the text data will be converted into lowercase and all the punctuations will be stripped.
- split- The input data is tokenized using 'whitespace'.
- Output sequence length- The fixed sequence length of each input text is defined using this parameter. If the length of the sequence is less than defined length then padding will occur. This parameter is set to 1500 after reviewing the size of each reviews.
- Output mode- This parameter represent the output of the text data. 'Int' is assigned to this parameter.

3.4 Model building

As discussed, this study has two phases of implementation. Initially, a LSTM model is built to predict the ratings from the input text. This is a multi label classification problem and is achieved through building an Input layer, an embedding layer and two layers of LSTM . The input layer of an LSTM model accepts data sequences presented as unprocessed text.This layer help in transforming the input data into a format that LSTM understand and use effectively. The input layer is defined by its data types and the shape of the input string. This layer receives a series of data points as input. The outcome of this layer is directed to a vectorization layer, where all initial raw inputs undergo filtration and transformation into tokens. These tokens will be represented as integers. The parameters were customised based on the review column such as vocabulary size, maximum sequence length, and output mode. The adapt method was utilized to create the vocabulary and fit the vectorizer for the user review data. The output of vectorization layer is fed into embedding layer. All natural language processing models use embedding layer as a fundamental layer. There are several steps involved in embedding layer including token representation, spacial representation. The output from previous layer includes only numerical representation of text. These numerical representation of tokens is transformed into dense vectors of a fixed size. These dense vectors are arranged in high dimensional space where similar words or tokens have same meaning or context will be represented closely. This layer again compress the high dimensional data into low dimensionality for efficient and fast computation. The process of initializing embedding weights involves assigning random values to them upon the creation of an embedding layer. Through the training process, these weights are fine-tuned via backpropagation. Once the model is trained, the word embeddings tend to capture similarities between words. The significance of the embedding layer is pivotal in this research. Notably, the embedding is derived from the available data instead of relying on pre-trained embeddings from external sources. The parameter used for this embedding layer were input dimension, output dimension and masking. The next layer used in this architecture is Long Short-Term Memory (LSTM) model which is used to identify patterns from sequential data in an effective way. The sequential patterns in the data is captured in LSTM layer using the word embedding sequence.The parameter used in this layer are the number of neurons and the activation function. The number of neurons is tweaked for obtaining appropriate model. This layer process the data at one step at a time which helps to learn and capture from previous data. Incorporating the dense layer into the model's architecture signifies the establishment of a comprehensive fully connected component. The knowledge distilled by the LSTM layer is harnessed within the dense layer to categorize comments or reviews within this model. The dense layer is configured with two key parameters: the number of units and the activation function. The count of the unique target variable is the total number of units used in dense layer.

3.5 Cosine similarity of embedding vectors

The degree of similarity between two vectors in a multi-dimensional space is quantified mathematically using the concept of cosine similarity. This approach use cosine of the angle between the vectors, to provide insight into their similarity. The model's second phase involves grouping the reviews. The assessment of similarity among these reviews is accomplished by computing the cosine distance utilizing embedding vectors. At the outset, each review, expressed as vectors, undergoes a comparison with another set of vectors to identify the degree of similarity. This likeness is established through a distance metric, with larger values indicating a closer alignment between the vectors.

4 Design Specification

The information was retrieved in its original raw text structure. In this framework, the primary outcomes consisted of the model's predictions for ratings and the identification of the closest neighbors utilizing the Scann algorithm. These text were analyzed and clustered together using the system design in Figure 3

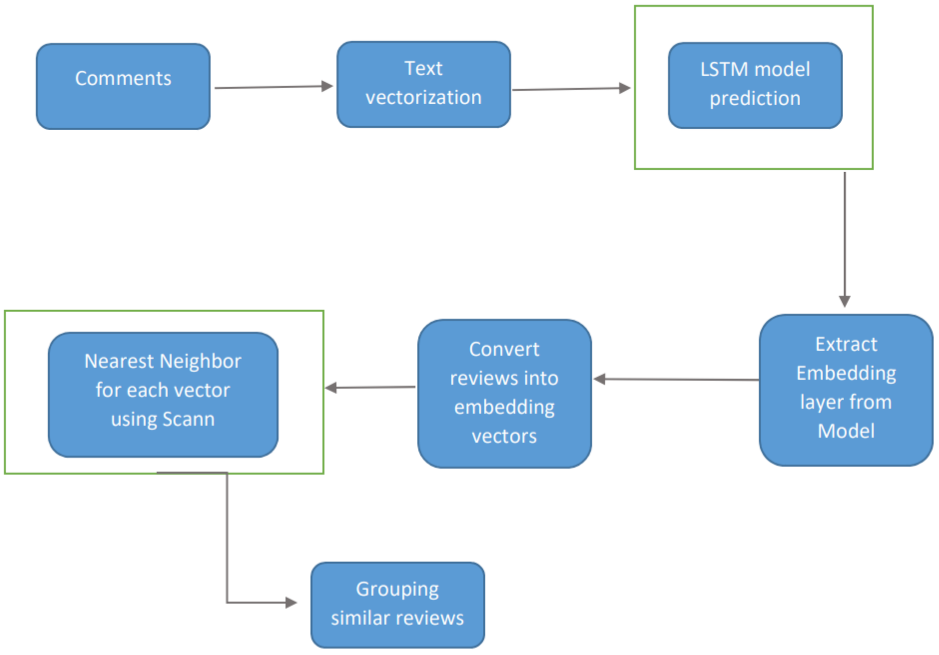

Comments
Text vectorization
LSTM model prediction
Extract Embedding layer from Model
Convert reviews into embedding vectors
Nearest Neighbor for each vector using Scann
Grouping similar reviews

Figure 3: Distribution of rating

5 Implementation

The proposed study is implemented using a deep learning framework which has Tensorflow and python libraries. The LSTM model which performs well with sequential data is used to predict multi label classification problem. The model building, extracting embedding vectors from model and finding cosine similarity required high hardware support. This research project was built on google colab pro+ with GPU accelerator attached to it. GCP is mounted to colab as storage for this research. The main packages used in this research are tensorflow which is used to build the model and scann to find cosine similarity between the vectors. The other packages like Keras, pandas, numpy, matplotlib were used. The latest version of tensorflow 2.12 is implemented for creating the model. An overview of implementation of LSTM model is provided below.

5.1 Data preparation

The data used for this study is extracted from Kaggle data source. The data talks about amazon product reviews which is provided by the user. The csv file downloaded from the kaggle is imported in colab using google.colab function. After reading the data into dataframe, the first step implemented was cleaning the data. The rows which has no comments and ratings were excluded. Many filters were introduced to clean the data including the removal of HTML tags and punctuation's. The target variable is analyzed to understand the distribution and to check the whether any imbalance data is captured. The target variable is plotted using matpoltlib package.The data required for TensorFlow-based LSTM model is prepared by converting Pandas DataFrame to a TensorFlow Dataset.TensorFlow provides a tf.data.Dataset API that allow the dataframe to efficiently load and preprocess it. The tensor_slice package accepts the description and rating columns as dictionary for conversion of dataframe to dataset.The transformation of the dataframe into a TensorFlow data set occurs in several batches. This process is facilitated by the creation of a preprocessing function prior to model construction.In order to improve the effectiveness of training and the generalization of the model, TensorFlow models, including deep learning models, use important techniques like batching and shuffling. The act of randomly rearranging the training examples is known as shuffling the data. The dataset need to be shuffled before the data is fed into the model because the predetermined order can cause biased output which reduce the efficiency of the model. Shuffling also aids in learning of more robust features in addition to the prevention of memorizing the order. The process of batching includes grouping various training example into batches. The weights of the model is updated on each batch which improves the stability and effectiveness of model training. The volume of each batch is determined by the batch size. The batch function partitions the dataset into multiple batches within the TensorFlow dataset. The optimized batch size value for this model is 32. Integrating both shuffling and batching functionality can improve convergence during training which helps the model not to fall in trap of local minima. It also let the GPU hardware which is used in the system to operate effectively. The preprocessing layer IntegerLookup in TensorFlow and Keras is frequently used to transform integer-encoded categorical features into dense vectors. The ratings were encoded using Integerlookup function in keras. The vocabulary of the input data is determined by adapt function present in IntegerLookup function. Adapt function will create a layer for data transformation. The encoded data of rating column will contain the integer encoded data after transformation.

5.2 Model Implementation for the prediction of ratings

The filtered data is passed to the sequential LSTM model to predict the ratings.The LSTM model starts with an input layer which accepts input data and keep the data ready for processing by the network's later layers. The parameter used for input layer is input shape and the dtype. The dtype is set to string. The input data received from input layer which holds text data is transformed into integer indices using the vectorizer layer. The next layer, embedding layer accepts input dimension parameter which is set to 100000, the output dimension parameter is set to 128. Mask zero parameter is set to true as the user reviews are of multiple lengths and padding is implemented. The initial LSTM layer is set with 180 neurons and the second LSTM layer use 280 units. The return sequence parameter is set to True as there is two LSTM layer used for this model. The activation function used in LSTM layer was tanh which adds value to train the model using GPU. The dense layer is set with 6 units for labeling the output. The activation function used in dense layer is softmax. An optimizer, loss function, and evaluation metrics were the three key parameters used for compiling the proposed model while training. The LSTM model architecture is shown in Figure 4 Verma (2023)

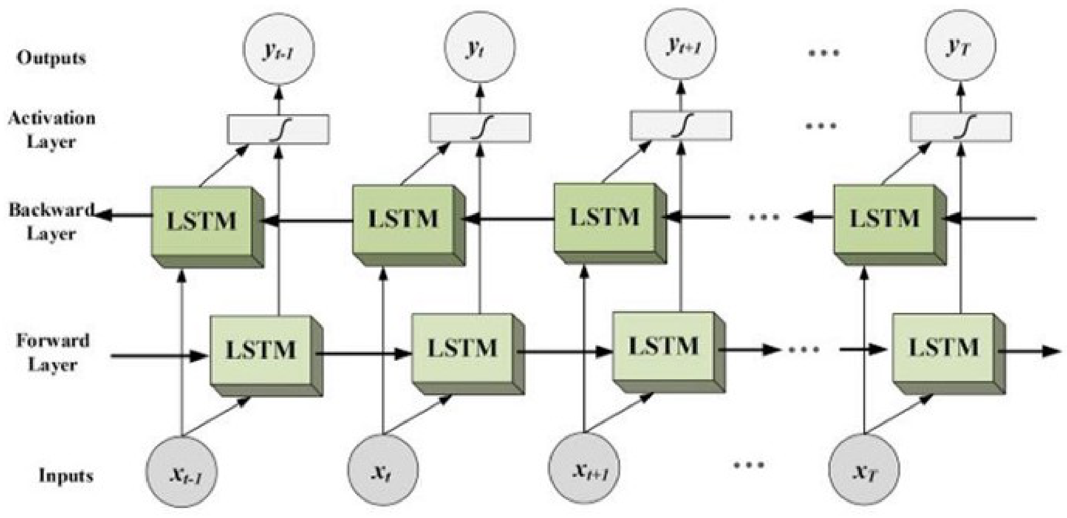


Figure 4: LSTM architecture

Adam is the optimizer used in this model with learning rate set to 0.001 after several tweaks. Loss is captured using tensflow API tf.keras.losses.categorical_crossentropy. Evaluation metrics is set to Accuracy. The data set was split into tarining set and test set in 80:20 ratio. Finally, fir method is invoked for training the model. After the model is trained, the embedding layer of this model is extracted to convert the text data into embedding vectors.The embedding layer is extracted using Model function which is part of Keras package. This is achieved after assigning name of the layer for both input and output parameters.

5.3 Clustering reviews

The clustering is achieved using Google's ScaNN also known as Scalable Nearest Neighbors. The vector representation of each review is saved in json and loaded further for finding nearest vectors. ScaNN is a research library created by Google for effective and scalable approximate nearest neighbor search. It can significantly speed up the process of finding related vectors in huge datasets and is designed to handle high-dimensional data. Every vector is matched against an array containing the other vectors, and the resulting distances between vectors are computed and presented in the distance parameter. To optimize the algorithm's performance, the algorithm extracts the top 20 closest neighbors along with their respective distances. Adjusting the distance threshold allows for customization in identifying the closest neighbors. In this study, neighbors with a distance exceeding 200 are filtered out. The collection of neighbors assigned to each review constitutes a cluster of reviews. scann.scann_ops.builder function is used to build and configure ScaNN indexes which used to search the nearest neighbor effectively. This function also accepts the parameter for setting the cluster size, and number of nearest neighbors and the count of hash tables. In this scenario, nearest neighbor of 10 is set while building scann model. A for loop is defined to identify the closest neighbor for every embedding vector. This is accomplished by utilizing the search feature within the scann model, with the nearest neighbor value is assigned with 20. The search in Scann is implemented using DOC_ID column which acts as primary key in the dataset. The target array is set with list of DOC_ID. Each target value is passed to filter_neighbours function to return 10 similar vectors which have a distance greater than 200.

6 Evaluation

For understanding and visualizing embedded vectors, the TensorBoard Projector is an excellent tool. As mentioned, after embedding, words which are having similar meaning will be placed closer to one another in low dimensional space, and this space can be viewed using tensorboard dashboard in Figure 5

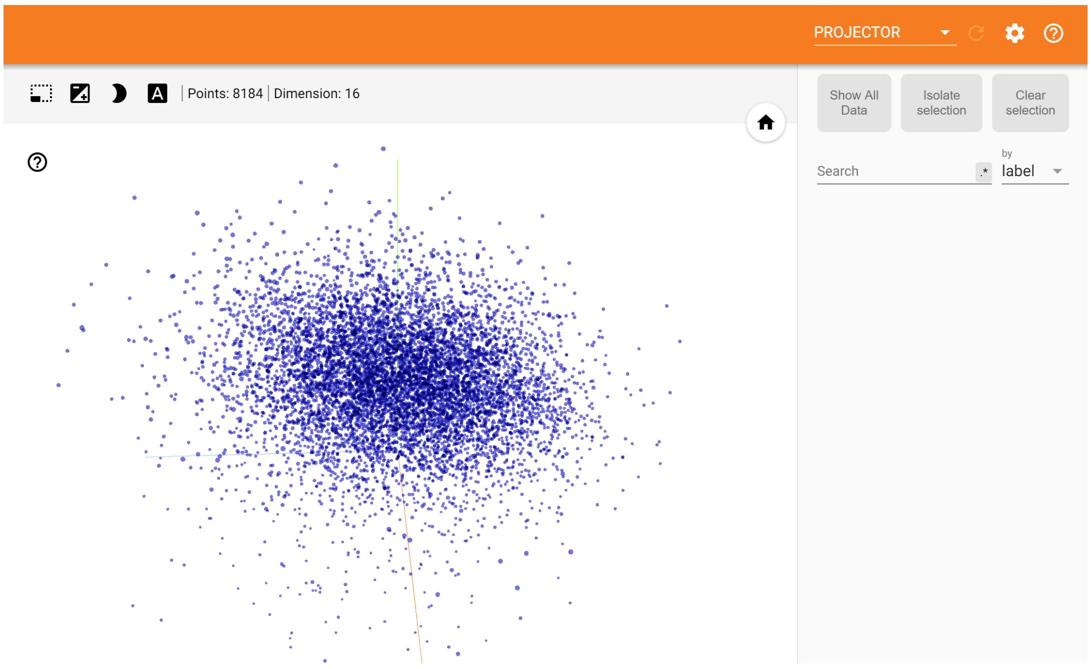


Figure 5: Word embedding projector

Performance and efficacy were two of the main factors this study took into account when developing an LSTM model. By developing quantitative metrics and qualitative analysis, this is accomplished. Multiple parameters were used to evaluate the LSTM model.

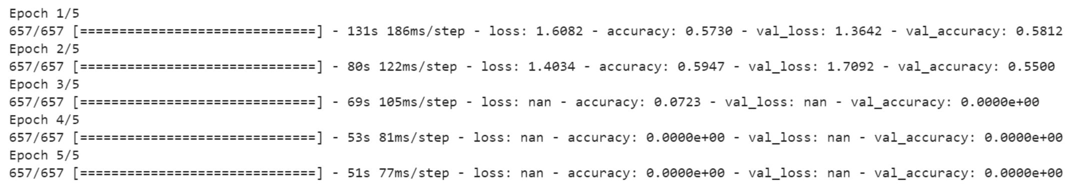
6.1 LSTM model built

LSTM model can be build using existing template. The quantitative and qualitative performance of the model were achieved after tuning the model. This is also called as hyperparameter tuning. The parameters used in the model were assigned with certain value and after evaluating the metrics these parameters were tweaked to achieve better accuracy. The performance optimization of the model is achieved by finding best hyperparameter combinations.

6.1.1 Learning rate

A key hyperparameter which determines the frequency of weight updation for the model is its learning rate. Learning rate indicate the level of convergence of the model either it is slow or fast. Choosing an appropriate learning rate can have a significant impact on the performance of the LSTM model.

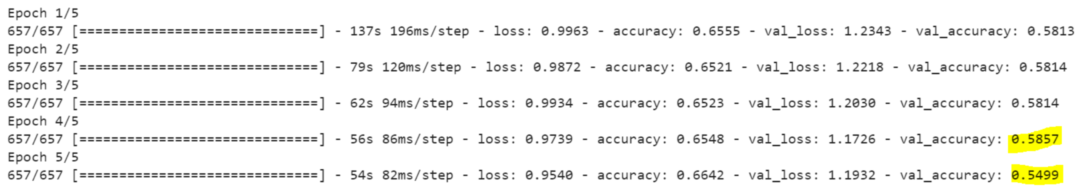
Initially model is chosen with learning rate=0.1 which shows fast convergence. Figure 6



```
Epoch 1/5
657/657 [==============================] - 131s 186ms/step - loss: 1.6082 - accuracy: 0.5730 - val_loss: 1.3642 - val_accuracy: 0.5812
Epoch 2/5
657/657 [==============================] - 80s 122ms/step - loss: 1.4034 - accuracy: 0.5947 - val_loss: 1.7092 - val_accuracy: 0.5500
Epoch 3/5
657/657 [==============================] - 69s 105ms/step - loss: nan - accuracy: 0.0723 - val_loss: nan - val_accuracy: 0.0000e+00
Epoch 4/5
657/657 [==============================] - 53s 81ms/step - loss: nan - accuracy: 0.0000e+00 - val_loss: nan - val_accuracy: 0.0000e+00
Epoch 5/5
657/657 [==============================] - 51s 77ms/step - loss: nan - accuracy: 0.0000e+00 - val_loss: nan - val_accuracy: 0.0000e+00
```

Figure 6: Learning rate =0.1

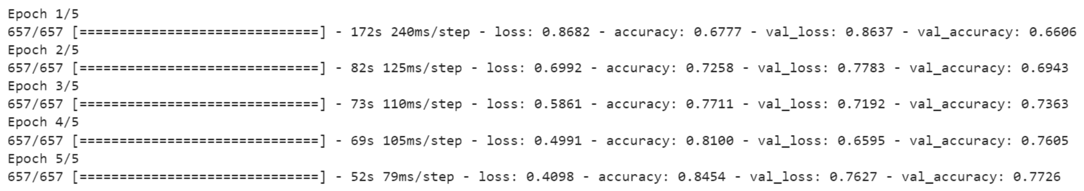
Next, model is chosen with learning rate=0.01 which shows an accuracy of 58%. The rate of convergence is slower than previous learning rate. Figure 7



```
Epoch 1/5
657/657 [==============================] - 137s 196ms/step - loss: 0.9963 - accuracy: 0.6555 - val_loss: 1.2343 - val_accuracy: 0.5813
Epoch 2/5
657/657 [==============================] - 79s 120ms/step - loss: 0.9872 - accuracy: 0.6521 - val_loss: 1.2218 - val_accuracy: 0.5814
Epoch 3/5
657/657 [==============================] - 62s 94ms/step - loss: 0.9934 - accuracy: 0.6523 - val_loss: 1.2030 - val_accuracy: 0.5814
Epoch 4/5
657/657 [==============================] - 56s 86ms/step - loss: 0.9739 - accuracy: 0.6548 - val_loss: 1.1726 - val_accuracy: 0.5857
Epoch 5/5
657/657 [==============================] - 54s 82ms/step - loss: 0.9540 - accuracy: 0.6642 - val_loss: 1.1932 - val_accuracy: 0.5499
```

Figure 7: Learning rate =0.01

Finally, model is chosen with learning rate=0.001. During the training, the accuracy spiked to 78%. Similarly, other parameters including number of neurons were also tweaked and the best accuracy was obtained with 180 neurons in first layer and 280 neurons in second layer. Figure 8



```
Epoch 1/5
657/657 [==============================] - 172s 240ms/step - loss: 0.8682 - accuracy: 0.6777 - val_loss: 0.8637 - val_accuracy: 0.6606
Epoch 2/5
657/657 [==============================] - 82s 125ms/step - loss: 0.6992 - accuracy: 0.7258 - val_loss: 0.7783 - val_accuracy: 0.6943
Epoch 3/5
657/657 [==============================] - 73s 110ms/step - loss: 0.5861 - accuracy: 0.7711 - val_loss: 0.7192 - val_accuracy: 0.7363
Epoch 4/5
657/657 [==============================] - 69s 105ms/step - loss: 0.4991 - accuracy: 0.8100 - val_loss: 0.6595 - val_accuracy: 0.7605
Epoch 5/5
657/657 [==============================] - 52s 79ms/step - loss: 0.4098 - accuracy: 0.8454 - val_loss: 0.7627 - val_accuracy: 0.7726
```

Figure 8: Final metrics

6.1.2 Compilation metrics

The sequential LSTM model used in this study is evaluated using different key metrics.

- Precision and Recall - Precision and recall is an important metrics for multi label classification model. This metrics is captured after segregating data into different groups. Precision is calculated by taking the ratio of correctly predicted positive values to the total predicted positive values. The precision is obtained with value 79%. Recall is the ratio of correctly predicted positive label to the total positive labels. This model has a recall value with 73%. Recall is also referred as measure of sensitivity.

- Accuracy- Another metric used to evaluate the performance of this LSTM model is accuracy. Accuracy is measurement of correctly predicted values. This metric is calculated taking the ratio of correct values predicted to total values available in the dataset. The training data accuracy was 85% and the test data accuracy obtained was 78%. The accuracy for the model is shown in Figure 9

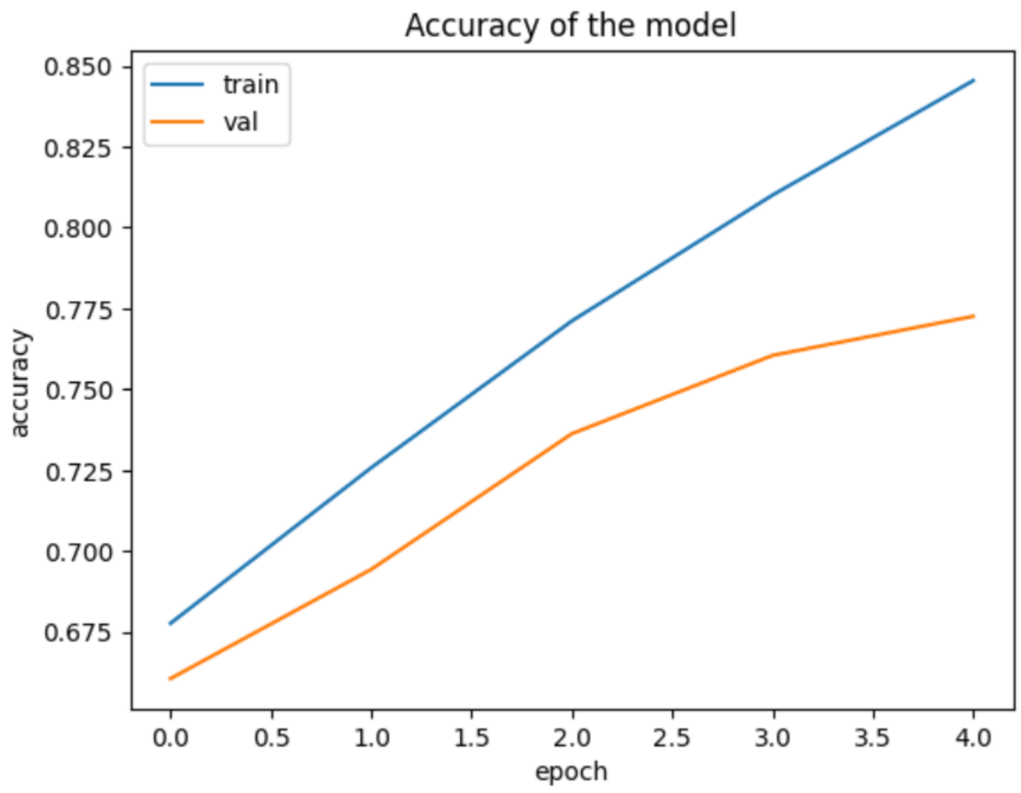


Figure 9: Model Accuracy

- Loss- The third metric calculated for this model is loss function. The loss function measures the difference between the predictions and the actual target values. The aim of the model is to minimize the loss value while training the model. If the value of loss is too low, this indicate the predicted label is similar to actual labels in the dataset. LSTM model can comply with different loss function and the issue which need to be addressed determine the loss function. In this study loss value observed is 0.40 for training data and for the test data the value obtained was 0.76. The loss for the model is shown in Figure 10

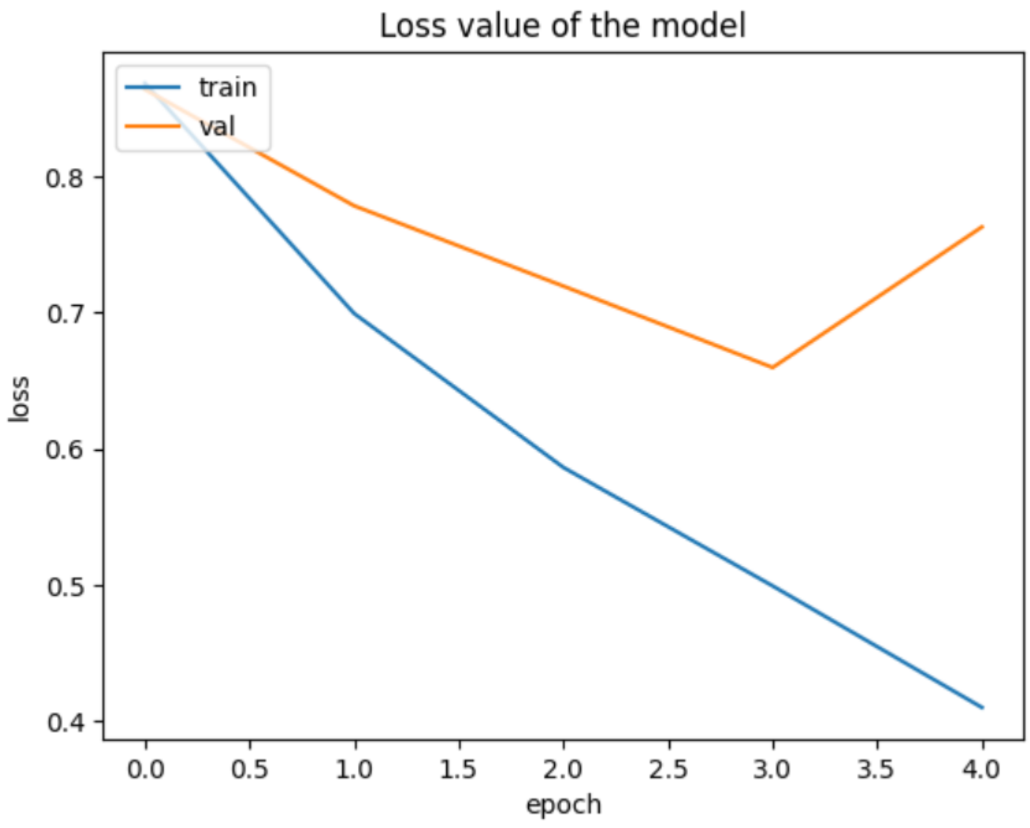


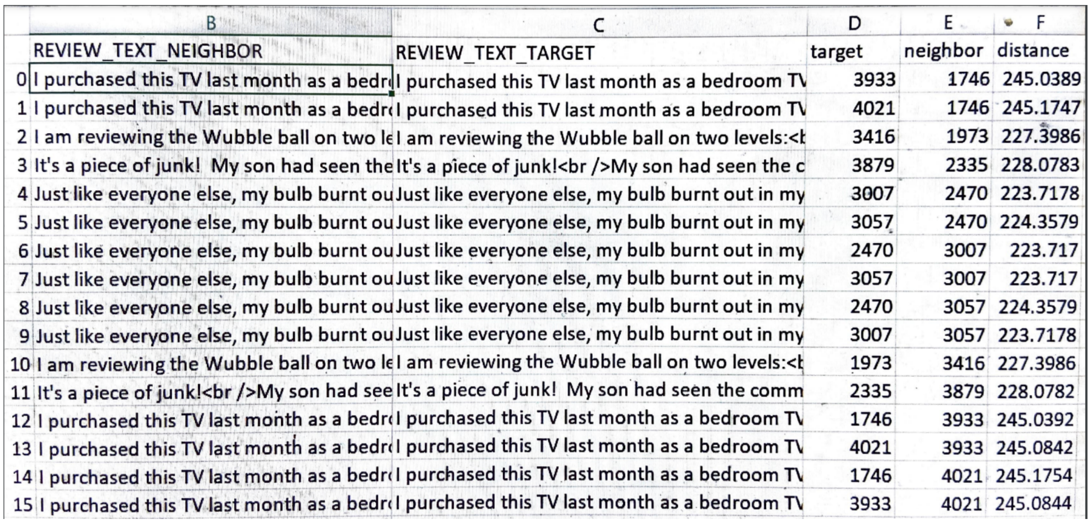
Figure 10: Model Loss

6.2 Grouping user reviews

The second phase of the proposed system is retrieval of the similar embedding vectors for each user review. The obtained output from Scann is validated using manual checks. The search were implemented using DOC_ID column. The output displayed in Figure 11 have a target column and neighbor column. The text looks quite similar with a distance greater than 220

6.3 Discussion

LSTM architectures excel in capturing sequential structures within textual information, rendering them well-suited to discerning the semantic interplay between words. Word embedding facilitates the aggregation of interconnected words, arranging them coherently within a vector space. The initial case study centered around assessing the performance of the LSTM model. The optimal model was established with a learning rate of 0.001 after meticulous parameter tuning. The sequential model yielded an impressive accuracy of 78%, accompanied by a loss value below 0.77. These metrics offer a compelling indication of performance, signifying the readiness of the initial system phase to generate embedding vectors for utilization in the subsequent model phase. The precision and recall obtained



	B	C	D	E	F
	REVIEW_TEXT_NEIGHBOR	REVIEW_TEXT_TARGET	target	neighbor	distance
0	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	3933	1746	245.0389
1	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	4021	1746	245.1747
2	I am reviewing the Wubble ball on two le	I am reviewing the Wubble ball on two levels:<b	3416	1973	227.3986
3	It's a piece of junk! My son had seen the	It's a piece of junk!<br />My son had seen the c	3879	2335	228.0783
4	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	3007	2470	223.7178
5	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	3057	2470	224.3579
6	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	2470	3007	223.717
7	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	3057	3007	223.717
8	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	2470	3057	224.3579
9	Just like everyone else, my bulb burnt ou	Just like everyone else, my bulb burnt out in my	3007	3057	223.7178
10	I am reviewing the Wubble ball on two le	I am reviewing the Wubble ball on two levels:<b	1973	3416	227.3986
11	It's a piece of junk!<br />My son had see	It's a piece of junk! My son had seen the comm	2335	3879	228.0782
12	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	1746	3933	245.0392
13	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	4021	3933	245.0842
14	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	1746	4021	245.1754
15	I purchased this TV last month as a bedr	I purchased this TV last month as a bedroom TV	3933	4021	245.0844

Figure 11: Similar users

for this model is 79% and 73% respectively. Rather creating embedding vectors from public dataset, the proposed study showcased to create vectors based on the dataset used in the study.

The subsequent stage of the proposed investigation involves identifying similar reviews through the utilization of embedding vectors. In this proposed system, the process of identifying comparable reviews for each individual review is based on a criterion where the computed distance between them is set to exceed a value of 200. Notably, when the distance between reviews surpasses the threshold of 220, the textual resemblance becomes particularly pronounced. Furthermore, it's observed that reviews falling within the range of 200 to 220 share a similar contextual essence, despite exhibiting slight variations in the vocabulary or words. Enhancing the grouping of reviews can be achieved by refining and fine-tuning the distance parameter appropriately. This could improve binding the reviews more efficiently. Previous researchers primarily focused on categorizing spam reviews, predominantly centered around detecting fabricated feedback. The proposed study goes beyond this by expanding the concept to identify malicious reviewers who systematically undermine products or businesses through the creation of multiple fraudulent profiles. The process of clustering users according to the semantic context of their comments enhances the overall business insights and implications. In previous research endeavors, the computation of cosine similarity for embedding vectors was executed using the API from the scikit package. However, this approach incurred substantial time for calculating vector distances, rendering its application less feasible for extensive datasets. This study introduces the utilization of the Google Scann algorithm, which offers expedited retrieval through efficient cosine similarity computations for embedded vectors.

7 Conclusion and Future Work

The goal of this study was to identify users who post fictitious reviews for their own gain. False reviews will damage the reputation of companies, goods, or services, which will decrease trust among real customers and make potential customers skeptical because of the prevalence of fake negative reviews. An exponential rise in false user account creation, majorly has the intention of posting negative comments. These comments can also affect search engine rankings and online visibility which will impact the business in bringing new customers. The research methodology employed a dual-stage approach, encompassing two distinct phases. Initially, a Long Short-Term Memory (LSTM) model was meticulously constructed to predict review ratings. This LSTM architecture comprised various layers, with particular emphasis placed on the embedding layer, a pivotal component in Natural Language Processing (NLP) models which is responsible for generating dense vectors. The model achieved a accuracy rate of 78%, underscoring the efficacy of the embedding layer. Subsequently, the Scann algorithm harnessed these embeddings to identify reviews exhibiting similarity. This process was facilitated through the utilization of the dot product API within the Scann algorithm, enabling the efficient comparison of vectors and thus contributing to the broader objective of review clustering.In the proposed methodology of this study,the identification of similar reviews originating from distinct fake user profiles becomes feasible through the process of grouping these profiles based on the similar characteristics find in their reviews. The limitation find in the existing system is that in the scenarios where a single user posts entirely disparate reviews across multiple profiles. In such cases, where the reviews lack substantial similarity, the proposed methodology may not effectively identify the association between the diverse profiles which is s constraint within the scope of this research.

Researchers can investigate a variety of options to improve the precision and efficacy of fake review detection methods with the expansion of technology and new techniques are developed. The potential expansion of the proposed research lies in its ability to broaden its functionality by incorporating the creation of a graph network after identifying these spam user profiles. This entails a more comprehensive exploration of relationships and interactions among users, forming a network structure that encapsulates various attributes and behaviors. By integrating a graph network, the research could delve into the intricate connections between legitimate and fraudulent user profiles, offering a deeper understanding of the underlying dynamics. This will facilitate to have more sophisticated analysis. This improved method could provide information about the larger context of spam-related activities, allowing for the identification of more complex patterns and boosting the overall efficacy of the research methodology.The another scope of extension is to integrate the proposed system with the online platforms of any business. This will enhance to have access to larger and more recent datasets and to gain a better understanding of the shifting tactics implemented by fake reviewers.

References

Cao, C., Li, S., Yu, S. and Chen, Z. (2021). Fake reviewer group detection in online review systems, *2021 International Conference on Data Mining Workshops (ICDMW)*, pp. 935–942.

Ibrahim, A. J., Siraj, M. M. and Din, M. M. (2017). Ensemble classifiers for spam review detection, *2017 IEEE Conference on Application, Information and Network Security (AINS)*, pp. 130–134.

Liu, P., Xu, Z., Ai, J. and Wang, F. (2017). Identifying indicators of fake reviews based on spammer's behavior features, *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 396–403.

Ramadhanti, N. R. and Mariyah, S. (2019). Document similarity detection using indonesian language word2vec model, *2019 3rd International Conference on Informatics and Computational Sciences (ICICoS)*, pp. 1–6.

Sandifer, A. V., Wilson, C. and Olmsted, A. (2017). Detection of fake online hotel reviews, *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 501–502.

Shahariar, G. M., Biswas, S., Omar, F., Shah, F. M. and Binte Hassan, S. (2019). Spam review detection using deep learning, *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp. 0027–0033.

Soyusiawaty, D. and Zakaria, Y. (2018). Book data content similarity detector with cosine similarity (case study on digilib.uad.ac.id), *2018 12th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, pp. 1–6.

Srivamsi, D., Deepak, O. M., Praveena, M. A. and Christy, A. (2023). Cosine similarity based word2vec model for biomedical data analysis, *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1400–1404.

Tang, H., Cao, H. and Li, J. (2021). A study on detection of fake commodity reviews based on ernie-bilstm model, *2021 International Conference on Computer Information Science and Artificial Intelligence (CISAI)*, pp. 1–5.

Thahira, A. and Sabitha, S. (2021). Graphical framework for review spammer group detection using metadata, *2021 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM)*, pp. 145–150.

Verma, Y. (2023). *Complete Guide to Bidirectional LSTM (with python codes)*, Analytics India Magazine, https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/.

Yao, C., Wang, J. and Kodama, E. (2019). A spam review detection method by verifying consistency among multiple review sites, *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 2825–2830.