Applied Soft Computing Journal 98 (2021) 106935

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

An ensemble-based hotel recommender system using sentiment analysis and aspect categorization of hotel reviews

Biswarup Ray, Avishek Garain *, Ram Sarkar

Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, West Bengal, India

ARTICLE INFO

Article history:
Received 23 June 2020
Received in revised form 5 November 2020
Accepted 18 November 2020
Available online 27 November 2020

Keywords:
Bidirectional Encoder Representations from Transformers
Categorization
Ensemble
Fuzzy
Hotel reviews
Random Forest classifier
Recommender system
Sentiment analysis

ABSTRACT

Finding a suitable hotel based on user's need and affordability is a complex decision-making process. Nowadays, the availability of an ample amount of online reviews made by the customers helps us in this regard. This very fact gives us a promising research direction in the field of tourism called hotel recommendation system which also helps in improving the information processing of consumers. Real-world reviews may showcase different sentiments of the customers towards a hotel and each review can be categorized based on different aspects such as cleanliness, value, service, etc. Keeping these facts in mind, in the present work, we have proposed a hotel recommendation system using Sentiment Analysis of the hotel reviews, and aspect-based review categorization which works on the queries given by a user. Furthermore, we have provided a new rich and diverse dataset of online hotel reviews crawled from Tripadvisor.com. We have followed a systematic approach which first uses an ensemble of a binary classification called Bidirectional Encoder Representations from Transformers (BERT) model with three phases for positive–negative, neutral–negative, neutral–positive sentiments merged using a weight assigning protocol. We have then fed these pre-trained word embeddings generated by the BERT models along with other different textual features such as word vectors generated by Word2vec, TF–IDF of frequent words, subjectivity score, etc. to a Random Forest classifier. After that, we have also grouped the reviews into different categories using an approach that involves fuzzy logic and cosine similarity. Finally, we have created a recommender system by the aforementioned frameworks. Our model has achieved a Macro F1-score of 84% and test accuracy of 92.36% in the classification of sentiment polarities. Also, the results of the categorized reviews have formed compact clusters. The results are quite promising and much better compared to state-of-the-art models. The relevant codes and notebooks can be found here.

1. Introduction

Initiation of the second generation of World Wide Web that is, Web 2.0 and the exponential growth of social networks, enterprises, and individuals have led to an excessive increase in the usage of the content available in these web resources which, in turn, help us to make highly informative judgments. Information processing from web resources also opens up many new research domains. For example, tourists consider checking past experiences and opinions of other travelers, available on the different web platforms, when planning their own vacations. This rich and diverse publicly available data can be used by giant tourist organizations as part of their on-field market research. This may range from carrying out polls or focusing groups of probable customers as future endeavors. But the diversity in opinions present in the textual data, provided by the users, gives rise to unwanted complexity as the processing of such huge data is a next to impossible task for humans. To this end, computer scientists provide some data-mining tools/algorithms which can help the user to extract relevant information from the vast amount of data. Taking into consideration a somewhat similar problem of creating a recommendation framework from opined texts available on the internet, this work focuses on developing a Hotel Recommendation System which can help both the tourism industry and individuals looking for hotels. In doing so, we use some advanced and comparatively new algorithms in the domain of textual data mining. Hotel Recommendation System based on Sentiment Analysis of the reviews is a very new research topic that has attracted the researchers due to its tremendous application in the hotel industry and tourism. It has multiple aspects, for instance, a review may talk about different categories such as location, room, and staff of a hotel. There are several factors that add complexity to study as well as retrieve useful information from such data. For example, the requirements of one particular

* Corresponding author.
E-mail addresses: raybiswarup9@gmail.com (B. Ray), avishekgarain@gmail.com (A. Garain), ramjucse@gmail.com (R. Sarkar).

https://doi.org/10.1016/j.asoc.2020.106935
1568-4946/

user vastly differ from another. Also the writing pattern of each reviewer is different from the other. For example, for the same hotel, different customers may give feedback in a completely diverse manner. Also the priority we give to some aspects varies a lot at a personal level. Some of us prefer food over hotel location, while some like to pay extra bucks for the window view. It varies both on gender as well as age basis. A millennial may prefer the availability of entertainment and spacious rooms. A typically old person may prefer better room service and cleanliness of rooms. Accordingly, we give the reviews. Another important factor is the size of the review set. Customers are keener to category wise personalized information found in the reviews and often use it as a basis for decision-making. We form an ensemble of different models of transfer learning using BERT and Random Forest classifier on different textual features of the reviews to classify the sentiments of the hotel reviews

In this work, we have focused on the Sentiment Analysis of the reviews crawled from the online Tripadvisor website made by online consumers. Then we have grouped the data into predefined categories. These categories like 'Location', 'Cleanliness', 'Service' etc. are the aspects that frequently recur in the review data, because topics often overlap with each other in real-world reviews. The remaining of the paper has been organized as follows. Section 2 provides a literature survey about the works already done on this topic along with a brief description of their performances. This is followed by Section 3 where we have discussed our motivation for the work and provided a brief decription of our contributions in the present work. Section 4 describes the datasets on which the proposed framework has been evaluated. The methodology that has been followed in designing our architecture is described in Section 5. This is followed by Section 6 where a detailed analysis of performance is shown. Finally, the concluding remarks are reported in Section 7.

2. Literature survey

Sentiment analysis indicates an area of natural language processing (NLP), computational linguistics, and text mining which aims to determine the emotions, personality, etc. of a writer analogous to specific topics. In recent years, many researchers have proposed various models on sentiment analysis of various topics of tourism, finance, social media, etc. Many types of research have been done in analyzing sentiments in the financial domain [1–4]. In 2020 Zhao et al. [5] proposed a BERT based sentiment analysis and key entity detection approach for online financial texts. By using a pre-train model, first, a sentiment analysis was done, and then key entity detection as a sentence matching or Machine Reading Comprehension (MRC) task was performed. Ensemble learning was also used to increase the performance of the method. Results showed that the performance of the approach was generally higher than conventional classifiers when evaluated on financial sentiment analysis and key entity detection datasets. Many research attempts have also been made on analyzing sentiments of texts or tweets extracted from different social media websites. For example sentiment analysis of recent tweets, found in social media, based on various topics such as Bhat et al. [6] on sentiment analysis of social media response on the COVID-19 outbreak, Manguri's model [7] on Twitter sentiment analysis on worldwide COVID-19 outbreaks. Also, sentiment analysis of Twitter data during critical events through Bayesian networks classifiers had been done by Ruz [8]. Due to the recent surge in the number of online communities, the methods of analyzing the online reviews written mainly by consumers to express their opinions, and the reviews on a specific product have fascinated many researchers. Many recent models had been used by various researchers for the analysis of such reviews. Some examples of such kind of research are as follows. Adaptable fine-grained sentiment analysis for summarizing multiple short online reviews collected from online Rotten Tomatoes (movie reviews), Amazon (shopping product reviews), etc. by Amplayo et al. [9]. Abdi [10] introduced an adaptable fine-grained sentiment analysis for the summarization of multiple short online reviews. The proposed model utilized online Naver movies dataset. Booking a hotel online has recently become a tedious task with thousands of hotels to choose from, for every destination. There are issues that come up quite frequently. For example, from the advertisement people may be convinced easily but they might not be aware of some hidden cost that the advertisers do not disclose online. So, customers who spent there, their feedback can be considered of much valuable importance. Motivated by the importance of these situations many researchers have made attempts to devise systems to solve this task and make the process easier for the people in need. The process of designing such review based systems have been worked through by various researchers with help of numerous techniques. One such famous technique is through sentiment analysis of the extracted online reviews made by various consumers in online websites. In 2020, Debraj et al. [11] devised a hotel review classification system. He used a machine learning-based approach for the classification where he summarized the hotel review information into a binary classification of positive and negative reviews. Opinion words present in reviews collected after feature extraction and pruning was used by the system. Each sentiment scores were categorized into different topics such as room, service, food, etc. Most common sentences in the review was generated to create a summary of the user-based queries in form of natural language. The efficiency of the algorithm was low (78.2% on average) which was obtained based on the average weighted F-score value calculated using Recall-Oriented Understudy for Gisting Evaluation (ROUGE). A machine learning-based sentiment analysis for analyzing the travelers' reviews on Egyptian hotels was proposed by Mostafa et al. [12]. The research proposed a traveler review sentiment classifier that analyzed the traveler's reviews on Egyptian Hotels and provided a classification of each sentiment based on hotel features. The sentiment model used three classification techniques: Support Vector Machine (SVM), Naive Bayes (NB), and Decision Tree (DT). For sentiment analysis. Classifier NB gave the highest accuracy 85% with a precision of 0.61 and a recall of 0.67 based on 11,458 sentiments. Kasper et al. [13] presented a system that collected reviews from the web and created classified and structured overviews of such comments and thus, facilitating access to that information. For each segment, the statistical polarity classifier yielded a positive or negative polarity value. The hybrid system utilized a combination of the polarity classifications from the statistical classifier and an Information extraction system. For a corpus of 1559 hotel reviews crawled from the web the system gave an accuracy of 0.67 and an accuracy of 0.83 with an F-measure of 0.81 for classification without neutral sentiments.

In 2016 Dey et al. [14] designed a framework to facilitate the quick discovery of sentimental contents of movie reviews and hotel reviews and their analysis. The process used statistical methods to find subjective elements and sentence polarity. The paper discussed two supervised machine learning algorithms: K-Nearest Neighbor(K-NN) and NB for the sentiment classification for a 2-class prediction task. For a training corpus of 4500 hotel reviews, the K-NN classifier gave an accuracy of 52.14% and the NB classifier gave an accuracy of 55.09%. Another such technique utilized by various researchers is of categorizing online reviews into various aspects and then performing aspect-based sentiment analysis. In the year 2017, a process on aspect-based sentiment oriented summarization of hotel reviews was proposed by Akhtar et al. [15]. The reviews and metadata were crawled from a website and classified into predefined classes mostly into some of the

common categories. These categories were the aspects that frequently recur in the review data. Then latent Dirichlet allocation (LDA) was applied to retrieve hidden information and aspects. Lastly, sentiment analysis on the classified sentences along with summarization was performed, thereby, building towards a Hotel Recommendation System. Tsai et al. [16] introduced a systematic approach that constructs classifiers to identify helpful reviews and hence, classifies the sentences in the reviews based on hotel features. Lastly, the sentiment polarities of sentences were analyzed to generate the review summaries. A Random Forest classifier was used for the classification task which achieved a 70% accuracy and 80% Area Under ROC Curve (AUC). In the year 2014, a novel aspect-based opinion mining technique was developed by Taylor et al. [17] for product reviews in the tourism domain. The work included the development of a generic architecture for an aspect-based opinion mining tool. Multiple expressions were used to denote the same attribute or component of a tourism product in reviews. The proposals were successfully implemented into a system and were used to tackle the issues in the Lake District tourism industry, in the south of Chile. The system achieved an F-measure of 0.9 for the sentiment analysis task by removing neutral reviews from the dataset, and could only extract 35% of the aspects from the reviews. The Nebular system proposed by Ayudhya et al. [18] classified comments gathered from various social network travel websites into predefined aspects and later analyzed comments into positive and negative sentiments. A list of words was used to map each word obtained from the data which was pre-processed initially into predefined aspects. Similarly, a list was also used to map a given word to the polarity. Thus, as a result, both the sentiments and the aspects of the reviews were grouped into a recommender system. The average accuracy of all modules was just over 85%. Albornoz [19] devised a joint model of feature mining and sentiment analysis for product review rating. The system first identified the features that were relevant to consumers when evaluating a certain type of product. The system then extracts from the review the opinions of users about different product features and quantifies such opinions. These are hence used to construct a *Vector of Feature Intensities*, which were the inputs to a machine learning model that classifies reviews into different rating categories. Over a dataset of 1000 hotel reviews from booking.com, the Logistic Regression classifier achieved a maximum accuracy of 71.7% for a 3-class prediction task.

Hotel review helpfulness has also been used as a measure by various researchers in formulating recommender systems. Hu et al. [20] proposed a novel multi-text summarization technique for identifying the top- k most informative sentences of hotel reviews. Both the content and sentiment similarities were used to determine the relationship between the two sentences. To identify the best sentences, the k-medoids clustering algorithm was used to partition sentences into k groups. Finally based on these groups the hotel information summarization was done. In the process proposed by Lee et al. [21], online hotel reviews were collected from the Tripadvisor website, and the usefulness of these reviews was comprehensively investigated from the review quality, sentiment, and characteristics of the reviewer. Prediction models to find helpfulness of a review were also developed using classification techniques. The techniques used to construct the model were (Decision Tree) DT, (Random Forest) RF, (Logistic Regression) LGR, and Support Vector Machines (SVM). On average, the method secured the highest accuracy for the Random Forest classifier with an accuracy of 82.7% and an F-measure of 0.826. Mahony et al. [22] described a system that was designed to recommend the most helpful product reviews to users. The system adopted a classification approach to harness available review feedback and to make learning a classifier that identified helpful and non-helpful reviews. Before classification, each review was translated into a feature-based instance representation. Each feature was derived from the categories: user reputation (R), social (SL), the top nine features for the dataset, which were used according to their information gain (IG). The results were divided into classes according to ratings. In the year 2016, Hu [23] proposed a model on the impact of review visibility, and interaction between hotel stars and review ratings on predicting hotel review helpfulness. Four categories of input variables were considered in the study: review content, sentiment, author, and visibility. Model Tree (M5P) was used as the prediction technique. The Part-Of-Speech Tagger (POS tagger) achieved about 97.3% token accuracy on average. Hu et al. [24] analyzed the importance of ratings, readability, and sentiments whether the reviews present online were manipulated or not. The paper proposed a simple statistical process to detect manipulation of online reviews. The analysis examined various information available in online reviews by combining sentiment mining techniques with readability assessments. The result showed that 10.3% of the products are subject to online reviews manipulation. Many researchers have proposed various models for the task of sentiment analysis. Many classification models widely used in the field of Machine Learning have been used for this task. Categorization of the reviews found in online websites written by the consumers has been used in developing the recommender systems. Also, various aspect-based sentiment analysis models has been proposed by researchers to design a hotel recommendation system.

3. Motivation and contributions

One of the major problems faced by the various researchers for designing a hotel recommender system is the lack of a properly labeled dataset. There are very few datasets containing hotel reviews. And even those datasets cannot be used for sentiment analysis or categorization task. There are no sentiment labels present in the dataset which are mandatory to train the dataset for the sentiment analysis task. This brings in the necessity for preparing a suitably labeled dataset containing the hotel reviews. From the literature survey, it has been observed that most of the researchers focus on the analysis of sentiments into two classes only i.e., positive and negative. However, there may exist some reviews which may contain a neutral sentiment. Hence, in the current work, we have considered this a 3-class problem. In this context, it is worth mentioning that conventional methods show low prediction accuracy for such a 3-class problem. The prime reason for such a lower accuracy is because there is an imbalance in the data caused by the smaller amount of neutral class data present in the dataset. BERT has been used by researchers earlier for sentiment classification, but the slower training process was a major problem faced by them. The emergence of TPU v3-8 (Tensor Processing Unit) has now ensured a much faster training process than before.

A user oftentimes wants sentiment information about a particular aspect or category belonging to a hotel. For example: a positive review for the food served by a particular hotel, thus leading to the need to categorize the hotel reviews according to various categories. In recent years, researchers have categorized reviews into some predefined sets by finding different similarity ratios among the reviews along with some manually extracted common terms for a particular category. However, the reviews may contain several misspells and typographical errors which limit the usefulness of such crisp similarity models.

Key contributions of this work are as follows:

- We have prepared a new sentiment tagged dataset which has been crawled from Tripadvisor trustworthy data sources, and it consists of reviews in the English language.

- We have proposed an ensemble of BERT [25] models as a pre-trained model and a Random Forest-based classification model which has worked on different textual features for the evaluation of sentiments on hotel reviews.
- Two phases of a BERT binary classification model giving higher weights to the neutral reviews are merged to get better results for the reviews containing neutral sentiments as they are lesser in amount than reviews containing other sentiments.
- Random Forest model uses different textual features such as Word2Vec embeddings, TF–IDF (Term Frequency–Inverse Document Frequency) of different words occurring more than a defined threshold value in the reviews.
- One major disadvantage of the BERT model is its slow training process, however, the usage of an advanced TPU v3-8 [26] architecture ensures much faster training.
- An ensemble of different models has led to a better and more accurate system beyond the reach of any single model.
- We have also categorized the reviews based on the aspects. In doing so, we have used a novel method by combining techniques of fuzzy string matching with cosine similarities of word vectors between the reviews and aspects, and their commonly recurring words.

4. Data

4.1. Data crawling

High-quality datasets related to hotel recommender systems are not publicly available as such. Especially the purpose of Sentiment Analysis and availability of properly balanced data are major challenges. That is why, in this work, we have been motivated to develop our own dataset based on our requirements. Also, a new dataset is always a valuable resource for the research community. The crawling of data was carried out using the Tripadvisor API. The website Tripadvisor has a huge database of reviews done by its users about hotels all around the world. All the reviews are geo-tagged leading to unique entries in the dataset and hence it makes crawling of data more organized and robust. This helps in avoiding duplication of data and thus supports data redundancy. The queries required to be fed by the users are domain and location codes to crawl location-specific review data. The raw data received were pre-processed and parsed employing regular expressions. The dataset can be downloaded from the following link: HotelReviewsdataset

4.2. Dataset characteristics

4.2.1. Metadata

The dataset consists of meaningful fields containing useful information about the hotels and the facilities provided by them from the reviewers' perspectives. It has 58,620 instances of data samples. The dataset [27] consists of the following fields:

- Review Id: Id of the review
- User Location: Location of user
- Reviewed Date: The date when the review was posted
- Hotel Name: Name of the hotel
- Date Of Stay: The check-in date of the reviewer
- Review Text: Review text posted by the reviewer
- Trip Type: Solo/with friends/with family/as couple/on business
- Value: Rating ranging from 1 to 5 for the value of the stay
- Cleanliness: Rating ranging from 1 to 5 for the cleanliness of the hotel
- Service: Rating ranging from 1 to 5 for service provided
- Location: Rating ranging from 1 to 5 for the location of the hotel
- Sleep Quality: Rating ranging from 1 to 5 for quality of sleep
- Rooms: Rating ranging from 1 to 5 for room quality
- Check-in/front desk: Rating ranging from 1 to 5 for reception service
- Business service: Rating ranging from 1 to 5 for other services like internet access etc.

4.2.2. Sentiment generation

Each of the fields containing review rating values has a range starting from 1 to 5. The more the rating value, the more positive is the reviewer's experience related to the field. Similarly lesser values signify negative experience.

Let $\mathcal{Z} = \{R_1, R_2, \ldots, R_n\}$ be the whole set of reviews in the dataset containing n review samples where R_i is a single review.

Now let R_i consist of a set of numeric values $R_i = \{r_1, r_2, \ldots, r_6\}$, where $1 \leq r_j \leq 5$ and represents the individual review value for the last 6 fields of the dataset.

Average review rating $R_{i,avg}$ is calculated using the formula mentioned in Eq. (1).

$$R_{i,avg} = \frac{\sum_{i=1}^{6} R_{ij}}{6} \tag{1}$$

Now the Sentiment value S_i for rating R_i is calculated as shown in Eq. (2),

$$\mathbb{S}_i = \begin{cases} -1, \text{ if } 1 \leq R_{i,avg} < 2.5 \\ 0, \text{ if } 2.5 \leq R_{i,avg} \leq 3.5 \\ 1, \text{ if } 3.5 < R_{i,avg} \leq 5 \end{cases} \tag{2}$$

These review texts and sentiment value pairs are used for Sentiment Analysis purposes. The dataset has been divided into train, validation, and test sets in the ratio 70:20:10 for this purpose.

For the categorization of the reviews, however, other fields are also necessary and no splitting has been done on the dataset.

5. Methodology

5.1. Pre-processing of review text data

Data pre-processing tasks are executed after the collection of data. The review texts that the reviewer writes consists of various types of words and their different forms. All these words and their various forms do not have any such significance for classification purposes. So, to generalize the data and preventing unnecessary use of computational resources, these texts need to be pre-processed. The text pre-processing tasks carried out for the Sentiment Analysis classification include:

- Removing stop words like 'the', 'a', 'this' etc.
- Extracting words from numeronyms (words made out of digits and letters as an SMS language form of original words) [28]
- POS tagging: Assigning one of the parts of speech to every word to define if it represents a noun, verb, etc. This information helps the model to capture some extra syntactic features from the texts.
- Lemmatization: Transforming each word into the root word it belongs to.
- Pascal casing of hash-tagged words [29]

5.2. Some additional features

Apart from the BERT model embeddings, some additional features are considered in the present work which are listed below.

- TF–IDF scores of the top 10 recurring words. We divided our training dataset into three parts depicting 3 documents, based on their review polarity. The TF–IDF score has two components:
  - TF: Term Frequency, which measures how frequently a term occurs in a document.

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document} \tag{3}$$

  - IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little or no importance.

$$IDF(t) = log_e \frac{Total\ number\ of\ documents(here = 3)}{Number\ of\ documents\ with\ term\ t\ in\ it} \tag{4}$$

- Vector outputs of the sentences when fed to Word2Vec. Some insights related to Word2Vec are as follows:
  - Distributed Representation: A type of representation in which the same features represent different concepts by using various combinations of the same. For example, 3 binary bits can either represent 3 things with it. Again they can represent 8 things using distributed representation as 3 bits can have 8 variations. In the context of Word2Vec, this implies that the feature/parameters for each word are a fixed length dense vector. Different variations of this same vector can represent different words. The feature space does not scale up with vocabulary size so it can handle a corpus of big size and learn from such a large dataset.
  - Approximate Softmax: Word2Vec is a neural classification model. It learns features or parameters for each word that allows it to predict surrounding words, and these models have a Softmax output layer. In Word2Vec, words are classes, so if we use a large corpus then it can have millions of classes. Word2Vec has a modified loss function which approximates the Softmax leading to the independence of training from a number of classes. This implies that Word2Vec can scale to millions of words.
  - Representation Learning: In Word2Vec, the features or parameters for each word are learned automatically i.e., they are not manually engineered. The model tries to learn the features or parameters that help it to predict surrounding words using the Gradient descent method.
- Counts of words with positive sentiment, negative sentiment, and neutral sentiment. [30] These counts help a lot in detecting the polarity of a review utilizing a linear dependency.
- Subjectivity score of the review text. The subjectivity is a real value within the range [0.0, 1.0], where 0.0 is very objective and 1.0 is very subjective. It has been calculated using the Natural Language Tool Kit(NLTK) library's TextBlob object.
- The number of question marks, exclamations, and semicolons in the review text.

The code snippets associated with the extraction of these features have been included in the Annex section.

5.3. Sentiment analysis

An ensemble of 2 BERT models along with a Random Forest classifier model based on textual features has been used for the Sentiment Analysis task.

5.3.1. *BERT model* 1

A binary classification based method built on BERT is used. BERT has obtained state-of-the-art performance on most of the tasks in NLP. As the dataset considered here consists of only text data in English, hence a BERT base cased model has been used. Both the *uncased* and *cased* models' results have been taken into consideration, and it has been noticed that the BERT base cased model has yielded better results for the same dataset uncased model. Also intuitively, it can be clearly noted that "WONDERFUL" may convey more in terms of sentiment than "wonderful".

The pre-trained BERT tokenizer for the BERT base cased model has been used to tokenize the reviews present as text in the pre-processed and structured dataset from the pre-trained BERT model based on the WordPiece model [31].

The pre-trained BERT models have been trained on a substantial corpus (Wikipedia + BookCorpus). The model shown in Fig. 1 is used for this purpose. Customization in terms of changing the loss function and optimizers are made in the model to get a better result. The steps for the BERT model as per Fig. 1 are:

- Special tokens [CLS] and [SEP] are added at the starting of the first sentence at the end of each sentence respectively.
- Segment embedding indicating different segments A, B is initiated to each token. Segment embeddings have a vocabulary of 2 and are similar to token embeddings.
- A positional embedding is initiated to each token to designate its position in the sequence.
- The entire input sequence is fed through the transformer Fig. 2.
- The output of the special token [CLS] is modified into a 2×1 shaped vector, an extra layer of classification is appended to the top of the Transformer output for the special token [CLS].
- The probabilities are calculated with a softmax function.

The transformer model used for finding the transformer outputs has been represented in Fig. 2. The figure explains the working of the encoder and decoder present in the transformer. The Encoder block has a single layer of Self-Head Attention followed by another layer of Feed Forward Neural Network. Whereas the decoder has an extra Masked Self-Head Attention. The encoder and decoder blocks consist of several encoders and decoders stacked over each other. The base model has a stack of 12 decoders. The produced output is a 768-dimensional vector. The model has 12 layers and 12 heads, resulting in a total of $12 \times 12 = 144$ distinct attention mechanisms. This stack of encoders works as follows:

1. The token, segment, positional embeddings produced from the input sequence are passed as input to the first encoder.
2. These embeddings are then transformed and send to the next encoder.
3. The outputs from the final encoder in the stack of the encoder are sent to all of the decoders.

Layer normalization normalizes input across features, unlike the batch normalization process which normalizes input features across the batch dimension. One word can have different meanings in a different context, and self-attention encode each word based on context words for the current word. The inputs are the element-wise sum of the token, segment, position embeddings.

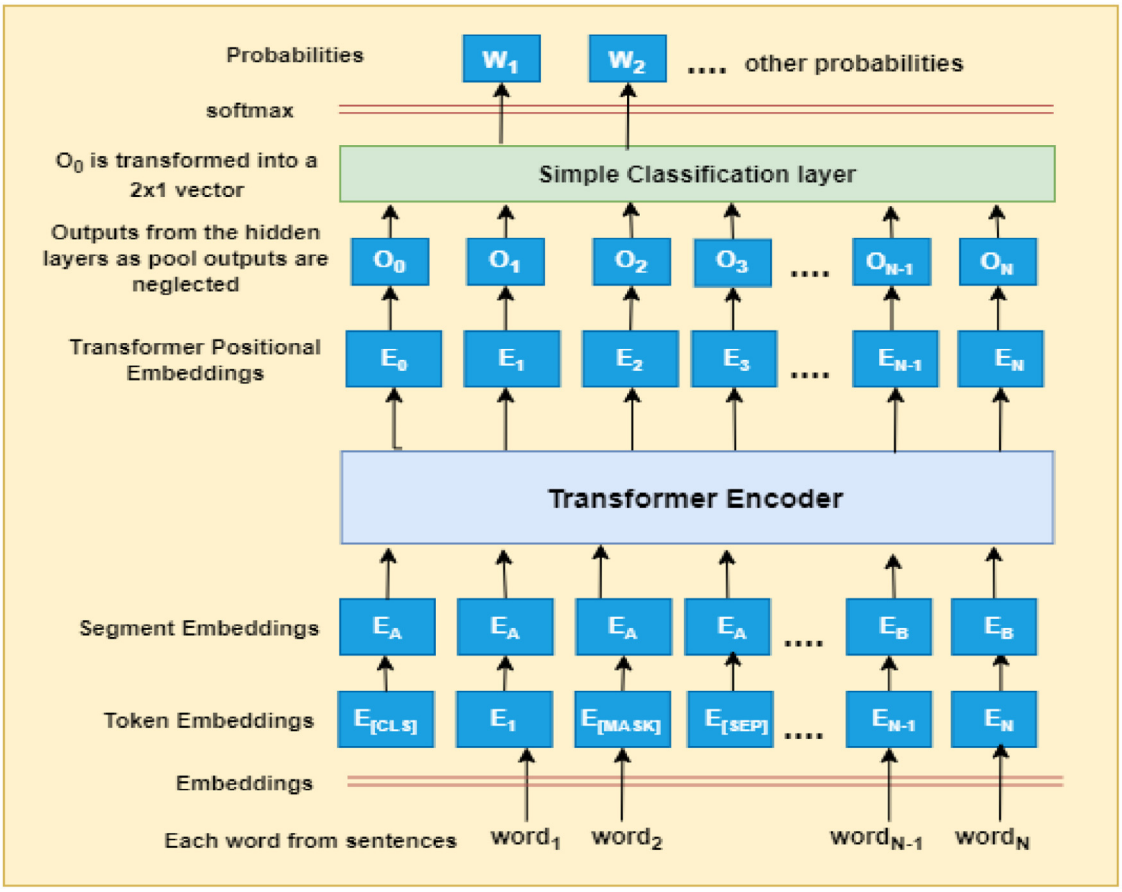

Fig. 1. Architecture of the BERT model used here for sentiment classification.

An example is taken from the reviews of the dataset and its token embeddings, segment embeddings, positional embeddings are shown in Fig. 3.

Token Embeddings layer transforms each Wordpiece token [31] into a 768-dimensional vector representation. The Segment Embeddings layer only has 2 vector representations. The first vector E_A is assigned to all tokens that belong to input 1 while the last vector E_A is assigned to all tokens that belong to input 2. Position Embeddings are the initial outputs given by the transformer. These representations are summed element-wise to produce a single representation. This is the input representation is passed to BERT's Encoder Layer. As the BERT model works with fixed-length sequences, hence, to find the maximum length, a plot for the token length of each review has been shown in Fig. 4.

From Fig. 4 it is noted that most of the reviews seem to contain less than 80 tokens. Hence, a maximum length of 90 is chosen. An Adam optimizer is used since it corrects the weight decay. An XLA optimizer [32] is also used which helps in improving the speed and memory usage. The loss function is chosen to be BCEWithLogitsLoss. A learning rate of $0.4 \times 1e-5 \times xm.xrt\_world\_size()$ has been selected and the model is trained for 15 epochs. The batch size has been set to be 32. For the training procedure, the dataset is divided into a ratio of 70:20:10 for training, validation, and testing purposes respectively. The training of the model is done in 3 phases. Firstly, for phase 1, only the reviews containing positive and negative sentiments are trained. Then, for phase 2, the reviews containing positive and neutral sentiments with neutral sentiments carrying a higher weight were trained. Finally, for phase 3, the reviews containing negative and neutral sentiments are trained with neutral sentiments carrying a higher weight than negative were trained. To get the predicted probabilities from our trained models, the softmax function is applied to the last hidden layer outputs from the, otherwise, the model returns output with logits for each label. The prediction probabilities for the neutral sentiment is calculated by merging phases 2 and 3. The probabilities from the two models are checked with a threshold t (0.78) and if found greater, it is termed as reviews containing neutral sentiment. All the other reviews are deemed as positive or negative sentiment based on the phase 1 model.

This framework is done to ensure a more accurate prediction of the reviews having neutral sentiment, since, the reviews having neutral sentiment are very low in number compared to the other sentiments. A higher weightage is given to these reviews by merging two phases of a binary classification model, thereby giving higher weights to the neutral reviews.

5.3.2. *BERT model* 2

A multi-label model is built using BERT. A BERT base cased model has been used in this method too. The pre-trained BERT tokenizer for the BERT base cased model has been used to get token representation from the pre-trained BERT model.

From the plot in Fig. 4, it is noted that most of the reviews seem to contain less than 80 tokens, but for better security, in achieving good results a maximum length of 90 is chosen. An Adam optimizer provided by Hugging Face is used since it corrects the weight decay. The same classification model as that shown in Fig. 1 is used. The loss function is chosen to be a cross-entropy loss function. XLA optimizer is not used for the

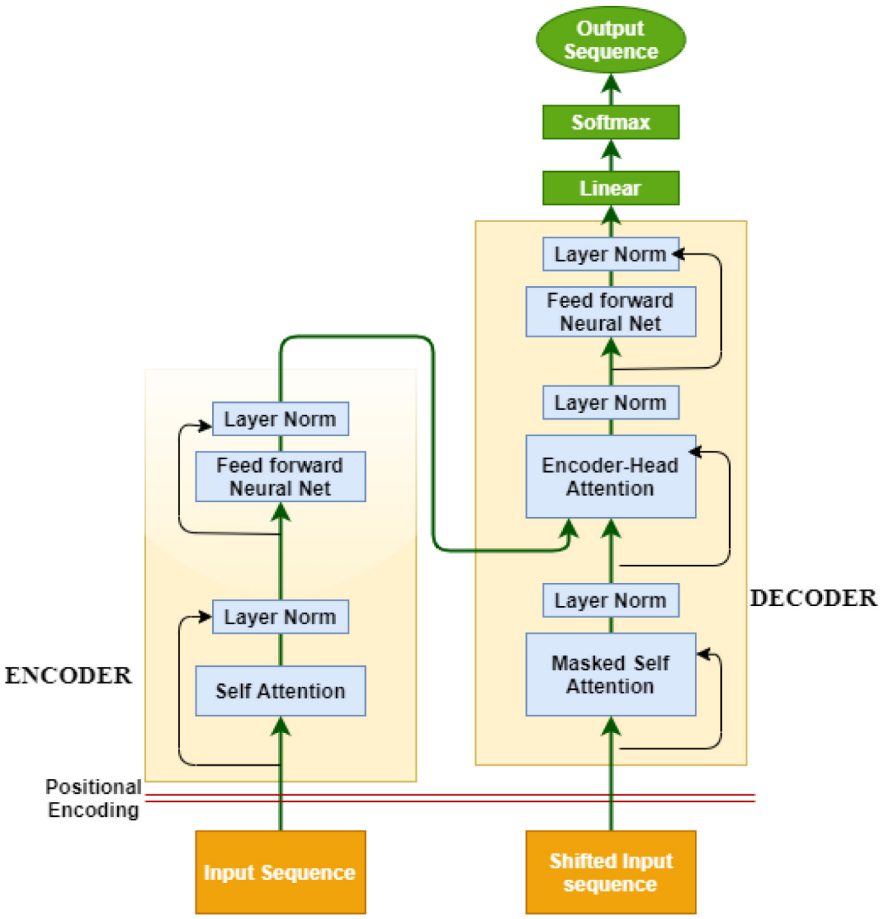

Fig. 2. Encoder and decoder present in the transformer of the BERT model used.

Fig. 3. An example is taken from the reviews of the dataset with its token, segment, positional embeddings.

optimization of the model. A learning rate of 2e-5 has been selected and the model is trained for 10 epochs. The batch size has been set to be 32. For the training procedure, a Dropout Layer for some regularization and a fully-connected layer for the output is used in the model. The Dropout Layer reduces overfitting in the model by preventing complex co-adaptations on training data. The raw output of the last layer is returned since the cross-entropy loss function must work. Hence, to get the predicted probabilities from our trained model, the softmax function is applied to the outputs. The scheduler got called every time a batch is fed to the model. Exploding of gradients is avoided by clipping the gradients. The prediction probabilities for each sentiment are provided by the outputs from the trained model (by applying the softmax on the model outputs).

5.3.3. Random forest model

A model is made using different text features extracted from each review for Sentiment Analysis using a Random Forest classifier. The vector representation of each of the words in the corpus

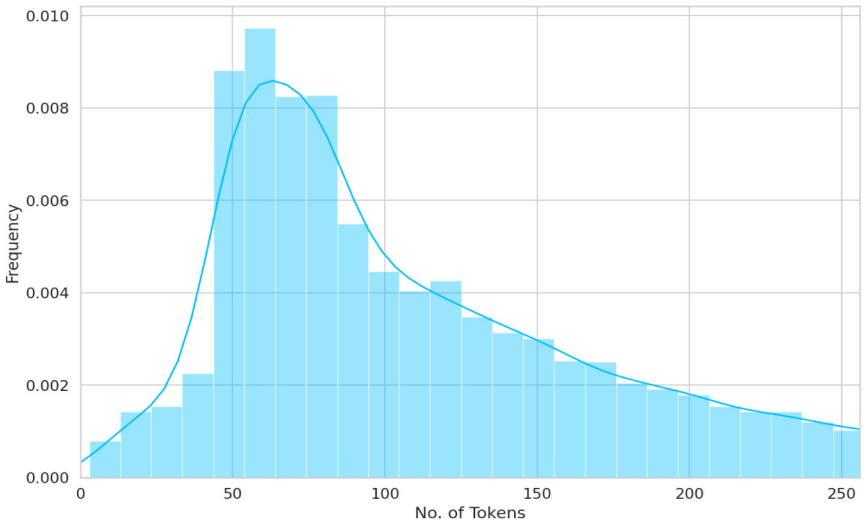

Fig. 4. A plot of token frequency vs. the number of tokens for a review.

is produced by using the Gensim module. The vectors of each word are the context in which the words appear (Word2Vec). Each of the text is also transformed into numerical vectors using the obtained word vectors. Thus, we train the Doc2Vec model by feeding in our review data. By applying this model to the reviews, we get the representation vectors that can be used as features for the model. In addition to the vector representation, the TF–IDF for the words that frequently appear in the text are also added to the feature list. The TF computes the number of times a word appears in the reviews, and IDF computes the relative importance of this word which depends on how many documents(here sets of reviews) the word can be found. TF–IDF is added as features to filter and reduce the size of the final output. The TF–IDF of all the W words present in a sentence is calculated by the formula:

$$\Delta tfidf(words) = \frac{1}{W}\sum_{i=1}^{W} \Delta tfidf(w_i) \tag{5}$$

The average polarity of each review along with the number of words is also added as features for the classifier.

For a review S, the average polarity of review is calculated by:

$$avg(polarity_{sub}) = \frac{1}{W}\sum_{i=1}^{W} polarity(w_i) \tag{6}$$

where $polarity(w_i)$ denotes the dominant polarity of w_i of S, as obtained from SentiWordNet. All the features which are extracted to be fed to the Random Forest classifier are listed in Section 5.2.

As the amount of neutral sentiments present in the dataset is low, the output probabilities are found by feeding the selected features to a Random Forest classifier with the weights to balance the classes i.e., it uses the values of labels to adjust weights as inversely proportional to class frequencies in the data. We have used this method as this is among the many commonly used methods which could be used to balance the classes.

Fig. 5 represents the working principle of the Random Forest model used for classification with the extracted text features.

Feature selection is performed on these features before classification. Features that have higher importance are used for the classification task. The top-20 features having the highest importance given by Random Forest classifiers can be seen in Table 1.

Table 1
Top 20 features and their importance obtained from random forest classifier.

Feature	Importance
doc2vec_vector_4	0.017564
doc2vec_vector_2	0.017304
doc2vec_vector_1	0.011866
doc2vec_vector_3	0.008727
doc2vec_vector_0	0.008614
word_great	0.007985
word_smell	0.007316
word_room	0.006835
word_dirty	0.006451
word_staff	0.006214
word_clean	0.006187
word_value	0.005705
word_facility	0.005570
word_poor	0.005418
word_rude	0.005300
word_bad	0.005040
word_comfortable	0.004955
word_pay	0.004642
word_location	0.004145
word_friendly	0.004096

5.3.4. Ensemble of the models

From the probabilities given by the 3 models, an ensemble based on a linear combiner is formed by using the validation accuracy as the weights. For a given set of 3 models with $j \in \{1, 2, 3\}$. A model $\mathbb{f}_j(c/i)$ with an estimate of the probability of class c of given input i, the ensemble probability estimate $\mathbb{F}$ is given by Eq. (5),

$$\mathbb{F}(c/i) = \sum_{j=1}^{3} \mathbb{w}_j \mathbb{f}_j(c/i) \tag{7}$$

where $\mathbb{w}_j$ is the weights provided to each model in the ensemble which is here validation accuracy.

An ensemble is used, to ensure that the final model uses the BERT embeddings generated and classified in the BERT models. Also, it ensures that the final model uses other textual features along with vector embeddings which are used for classification in model 3.

The proposed architecture for the ensemble of the 3 models is shown in Fig. 6

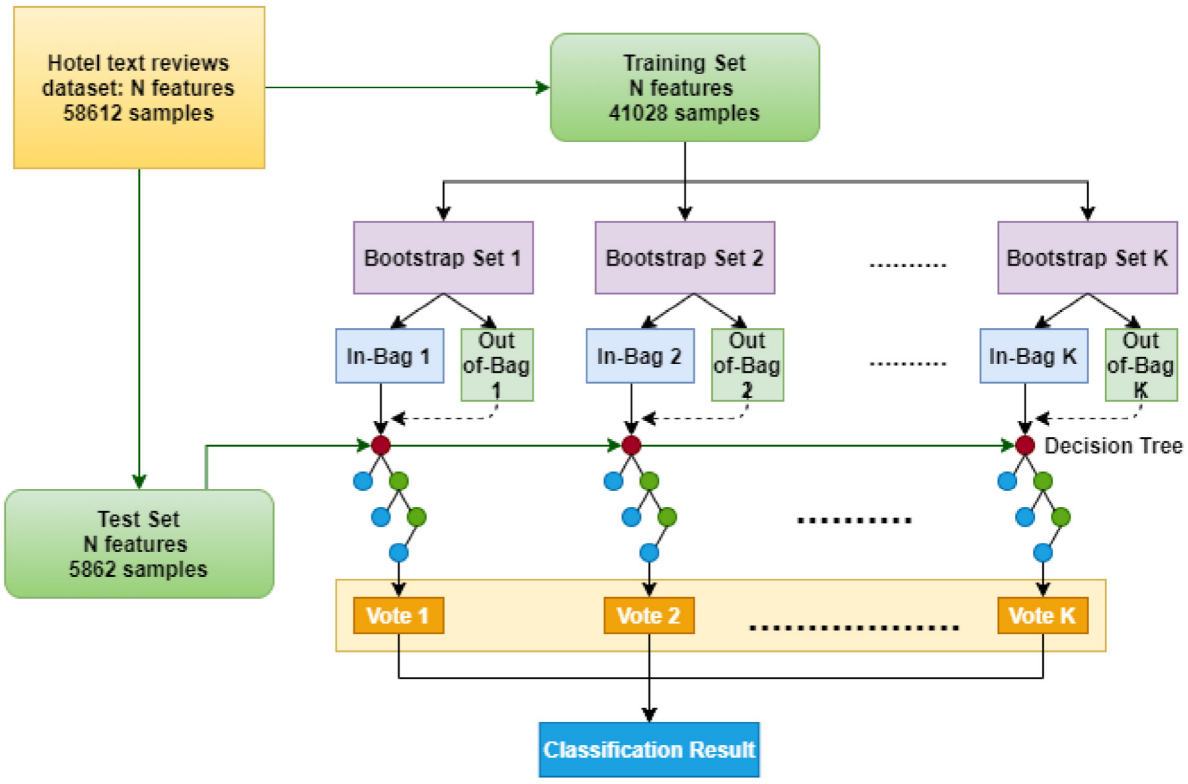

Fig. 5. Model 3 of our ensemble using Random Forest applied for classification with the extracted text features.

5.4. Category division based on aspects

This step involves grouping the reviews into some predefined categories. These categories are the aspects that frequently recur in the review data as shown by the word cloud in Fig. 14. Zhan [33] proposed an effective approach to discover these categories and their corresponding index terms for categorization. This method outperformed other conventional methods.

Hence, this approach has been implemented for the extraction of nouns present in each of the review sentences where each term frequency value is measured using the following formula:

$$Tf(k_j) = \sum_{i=1}^{|S|} fr(r_i, k_j) \times log \frac{|S|}{N(k_j)} \tag{8}$$

where, S is the set of reviews with length $|S|$, $fr(r_i, k_j)$ is the frequency of term k_j in the review r_i and $N(k_j)$ is the number of reviews when k_j emerges in S. Representative nouns are found by the ranks of their term frequencies.

Finally, a set of hotel categories is found by the highly-rated nouns. The terms that are significant to each of the hotel categories are inspected and gathered, and the index sets for different hotel categories are constructed. The sets of these categories along with their commonly recurring words are listed in Section 6.3.

The categorization of the reviews is done by finding the similarities among the reviews with each element in the index set of different categories through 2 different processes:

- The fuzzy string matching method is used to calculate the similarity ratio between each review with the index terms in each category. This matching process uses Levenshtein distance to calculate the differences between the words present in each review. The Levenshtein distance, a metric to measure how apart two sequences of words are, between two sequences s and k is:

$$\mathbb{l}even_{s,k}(i,j) = \begin{cases} max(i,j), \text{ if } min(i,j) = 0 \\ \min \begin{cases} leven_{s,k}(i-1,j) + 1 \\ leven_{s,k}(i,j-1) + 1 \text{ otherwise,} \\ leven_{s,k}(i-1,j-1) + I_{(s_i \neq k_j)} \end{cases} \end{cases} \tag{9}$$

where $I_{(s_i \neq k_j)}$ is an indicator function which is equal to 0 when $a = b$ and equal to 1 for other cases. $leven_{s,k}$ gives the distance between the first i characters of s and the first j characters of k. The similarity ratio $leven_{ratio}$ can then be calculated by:

$$leven_{ratio} = \frac{(|s| + |k|) - leven_{s,k}(i,j)}{|a| + |b|} \tag{10}$$

where $|s|$ and $|k|$ are the lengths of the sequences s and k respectively.

The average of the similarity values between the words present in each review with the index set elements is calculated to find the fuzzy similarity value for a review to belong to a particular category. The fuzzy string matching method has been used as the keywords present in the list of categories may not be present as it is in the reviews. Since the reviews have been extracted from the websites, misspells and typographical errors are common. To address these issues, a fuzzy string matching technique has been used in the proposed work.
- The cosine similarity is calculated between each word vector present in a review with the index terms in each category. The cosine similarity Cos_{sim} between two-word vectors can

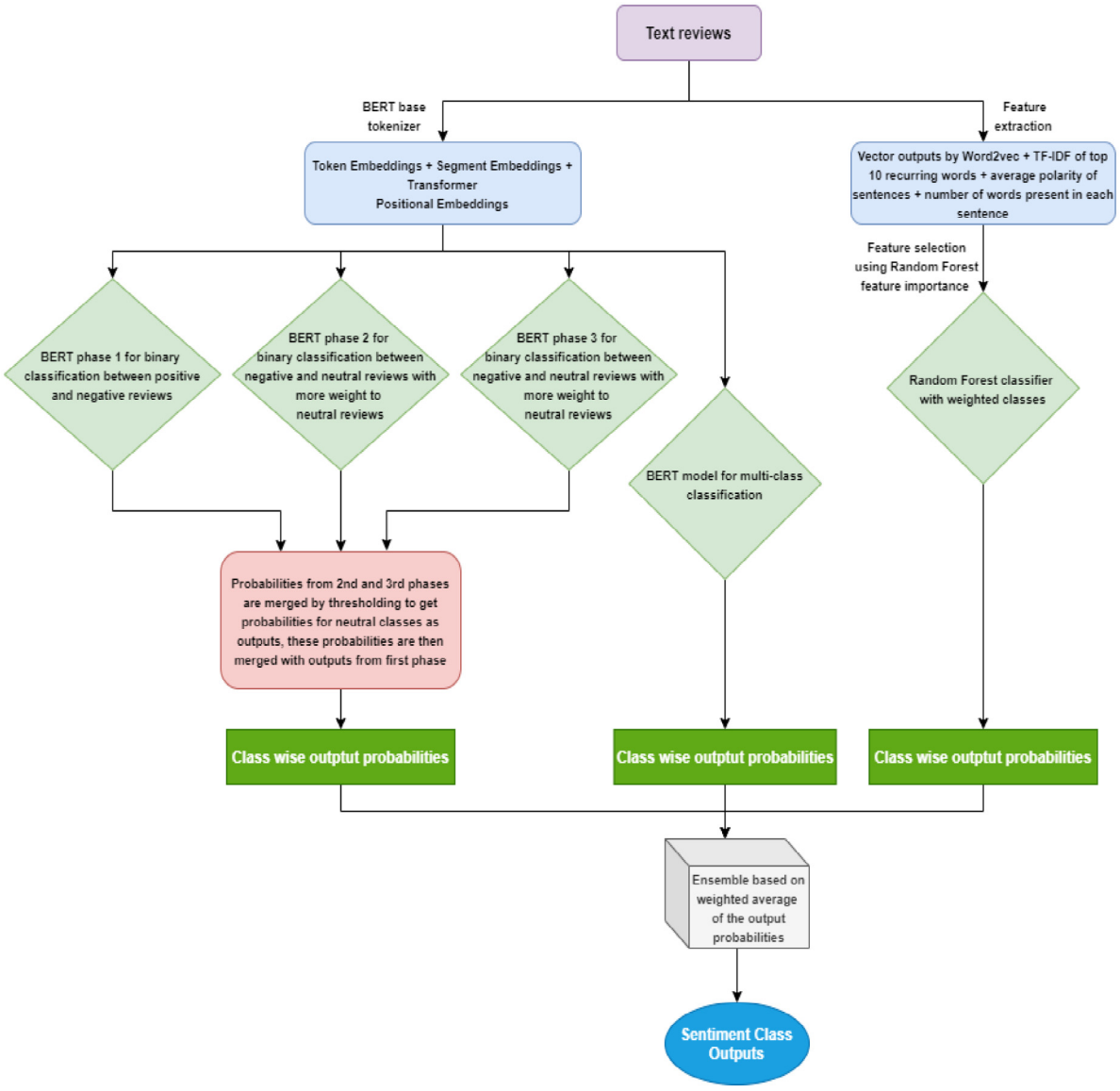

Fig. 6. Architecture for the proposed Sentiment Analysis model.

be defined as:

$$Cos_{sim}(\vec{A}, \vec{B}) = \frac{(\vec{A} \cdot \vec{B})}{|\vec{A}| + |\vec{B}|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} + \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (11)$$

The angular similarity $angular_{sim}$ is then calculated by:

$$angular_{sim} = 1 - \left(\frac{cos^{-1}(Cos_{sim}(\vec{A}, \vec{B}))}{\pi}\right) \quad (12)$$

The $angular_{sim} \epsilon [0, 1]$. Hence, the use of $angular_{sim}$ ensures that the similarity values lie in the range of [0, 1].
The average of each term and each word is calculated to find the cosine similarity value for a review to belong to a particular category.

The average of the resultant similarity values from both the methods falling in the range [0, 1] is then calculated and the category for which the maximum value is achieved is taken into account. The reviews are hence classified into the category for which they have the maximum value.

An example of a review with an average of its similarity value for each category is given in Fig. 7.

The overall model architecture for the review categorization process is represented in Fig. 8

5.5. Hotel recommendation system framework

In the architecture for the proposed hotel recommendation system as shown by Fig. 9, at first unstructured data are crawled and pre-processed to form a clean and structured dataset from reviews found in online hotel review websites. Features are then extracted from this dataset. Sentiment classification is done on the reviews with the help of an ensemble of BERT and Random Forest classifier models. Then with the help of similarity values between the reviews and an index set of predefined categories, the reviews are categorized.

For any input query from a user, searching is performed based on the preferred location for the hotel given as an input query by the user. Then the results for this search are sorted based on the predefined categories of (cleanliness, value, etc.) and are shown

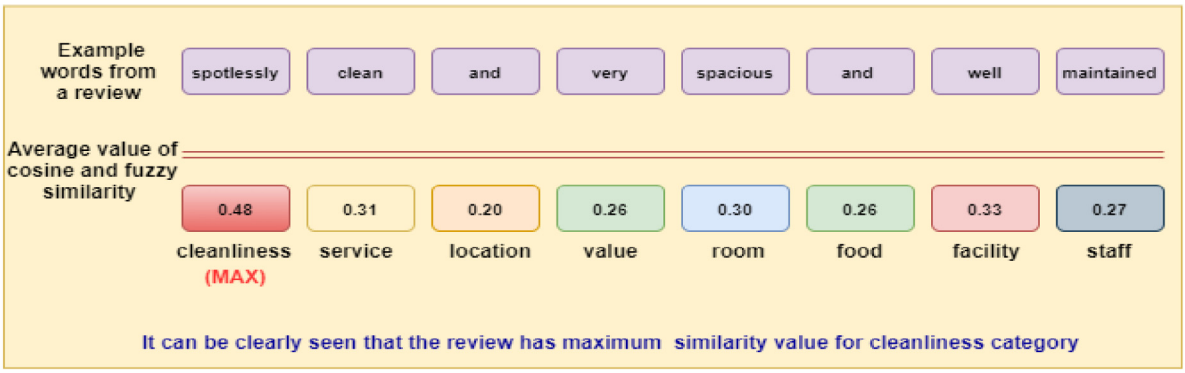

Fig. 7. An example review with its similarity value for each of the aspect based category. The similarity value is the average of fuzzy string matching value and angular similarity value. The maximum similarity value among the categories in the category selected for the particular review and the maximum value among the categories is indicated.

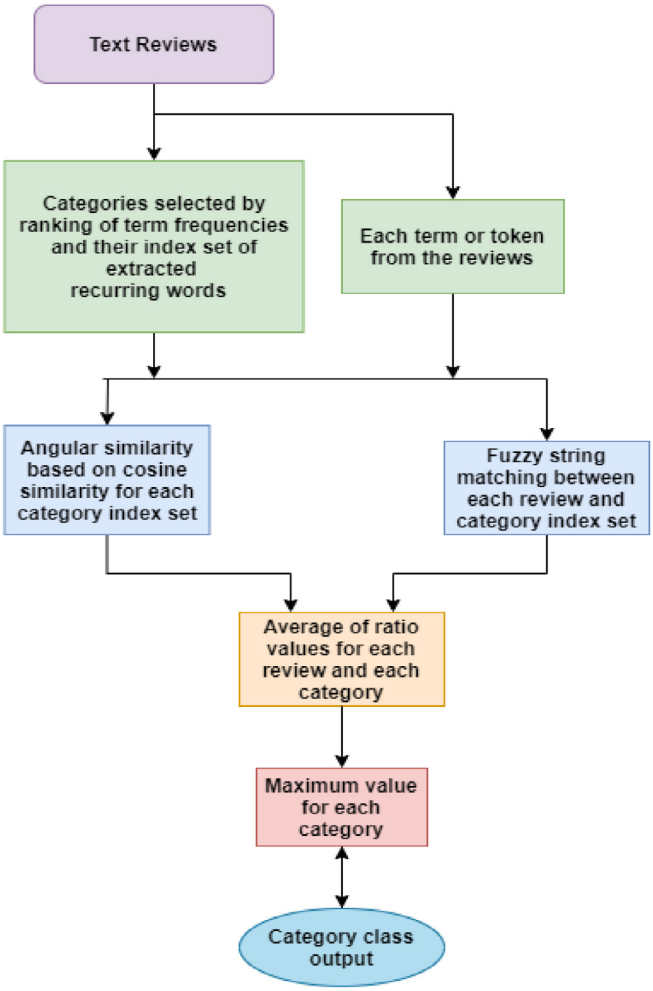

Fig. 8. Architecture for the proposed review categorization process.

as initial outputs. Polarities (sentiments) for these output lists of reviews and hotel pairs are then checked. Finally, an output of a particular hotel with reviews of particular polarity (sentiment) as per user query is produced.

Table 2
Training and testing time taken by different models for the sentiment analysis task.

Training time (10 epochs)		Testing time		
BERT with TPU	BERT with GPU	BERT with TPU	BERT with GPU	Ensemble
3526.6 s	8283.6 s	137.7 s	176.5 s	206.2 s

6. Results and analysis

In this work, we have proposed a hotel recommendation system based on Sentiment Analysis and categorization of hotel reviews. We have prepared our own dataset by crawling data, carried out using the Trip advisor API. The crawled dataset consists of 58 612 reviews. The results for both the processes of Sentiment Analysis and review categorization are discussed in detail. The libraries used for data crawling are urllib, socket, and contextlib. The gensim [34] library's efficient Word2Vec functionality has been used for generating vectors from words. The PyTorch [35] library has been used to implement the architecture explained in the above sections.

All of the training for the BERT models have been done in a TPU v3-8 [26] architecture. TPU is an artificial intelligence (AI) accelerator application-specific integrated circuit (ASIC) developed by Google. It is designed to accelerate deep learning tasks developed using the Tensorflow framework. All the other aforementioned models other than the BERT models have been trained using NVIDIA K40 GPUs. The fuzzywuzzy python library [36] has been used for fuzzy string matching in review categorization.

6.1. Computational analysis and feature settings

The training time for the different BERT models and testing time for the models are shown in Table 2. A ratio of 70:20:10 of Train:Validation:Test has been considered.

From Table 2 it can be clearly observed that the time taken for both training and testing for a BERT model with GPU is much larger than that of a BERT model with TPU. Hence, by using a TPU model we could reduce the computational time of the BERT model by a significant amount. To obtain the best combination of the feature settings. The models have experimented with different combinations of feature settings. The accuracies given by the different feature settings have been listed in Table 3. From the table, it can be observed that the accuracy given by the combination of all the features provides the best result for the

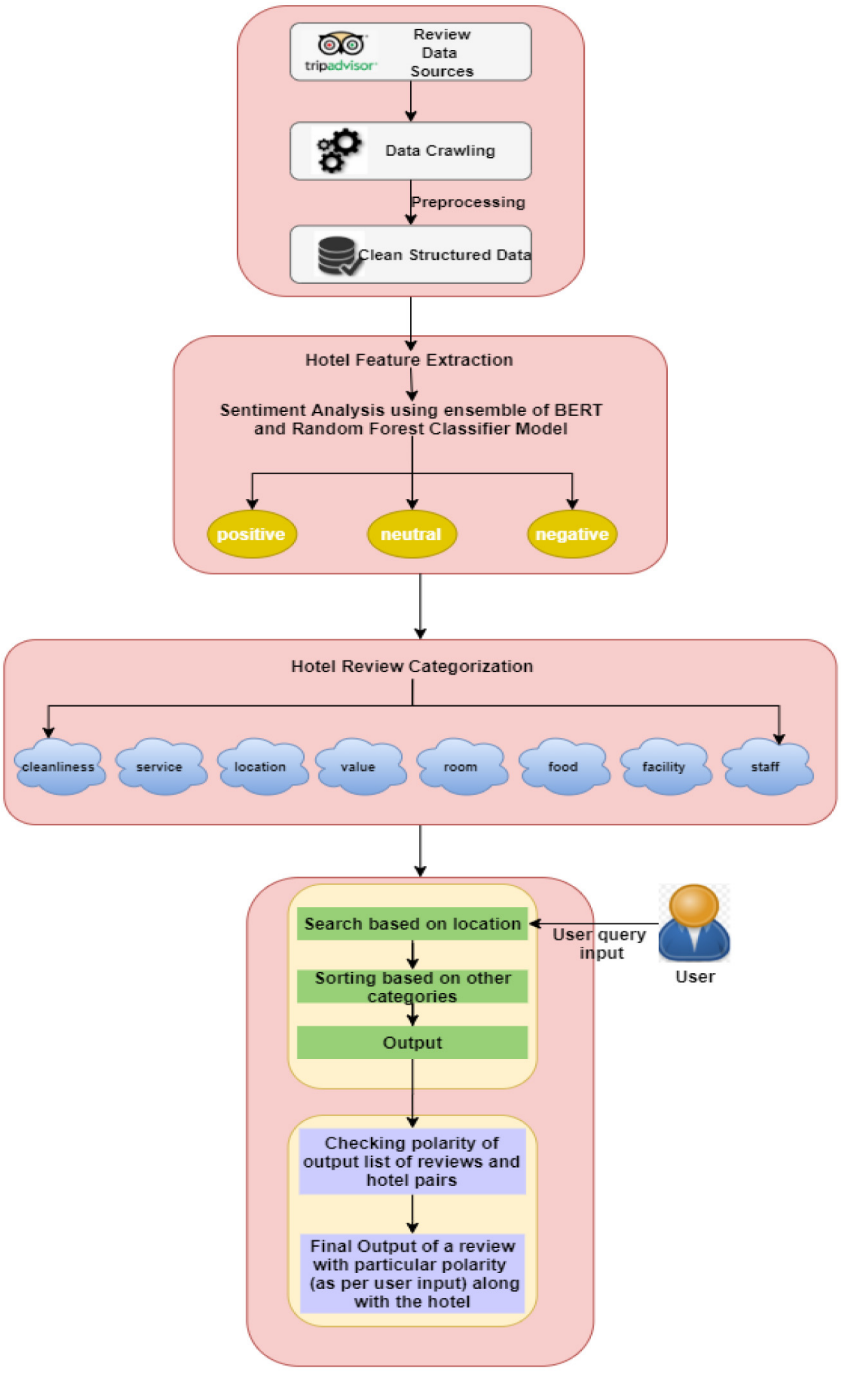

Fig. 9. Architecture for the recommendation system.

sentiment analysis task. Thus keeping this in accordance, we have used a combination of all the features thus providing us with an accuracy higher than any other setting.

6.2. Sentiment analysis

The results of the proposed method for Sentiment Analysis are compared to some state-of-the-art methods used for Sentiment Analysis such as Recurrent Neural Networks (RNN) [37], Gated Recurrent Units (GRU) [38], Long Short Term Memory (LSTM) [39] and Bidirectional LSTM (Bi-LSTM) [40] for the same dataset.

Table 3
Different feature settings and the corresponding accuracy for sentiment analysis task.

Feature setting	Accuracy
Only Bert	90.36
Bert and Word2vec feature	91.77
Bert and TF–IDF feature	91.52
TF–IDF and Word2vec feature (without Bert)	88.83
All the features (Proposed model)	92.36

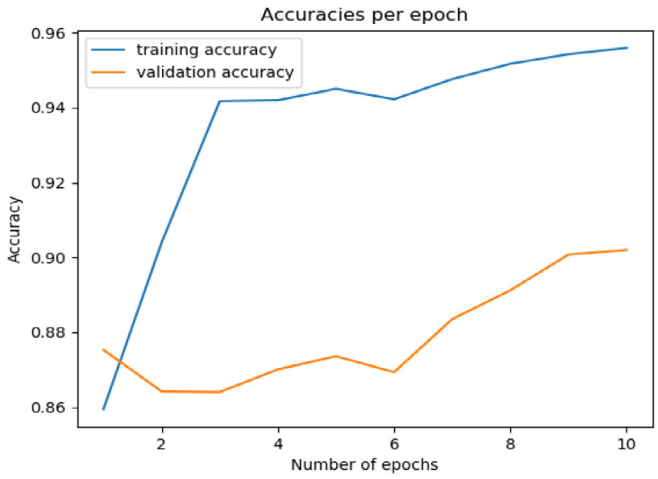

Fig. 10. A plot between the training and validation accuracies per epoch of BERT model 2.

Table 4
Comparison of the proposed model with some state-of-the-art models in terms of training, validation, and test accuracies.

Model	Train (%)	Validation (%)	Test (%)
RNN	84.75	84.01	86.00
GRU	87.57	85.57	90.00
LSTM	88.51	84.70	89.00
Bi-LSTM	87.98	85.43	89.00
Our model	**94.35**	**92.79**	**92.36**

For the evaluation of the performance of the model, some standard metrics such as accuracy, sensitivity, specificity, precision, recall, and F1 score are considered. For a 3-class problem as given here, with n observations, the accuracy can be measured as:

$$accuracy = \frac{1}{n}\sum_{i=1}^{3}\sum_{x:f(x)=i} I(f(x) = \hat{f}(x)) \tag{13}$$

where I is an indicator function which returns 1 if the classes match and 0 otherwise and $f(x)$ is the output function for the classification.

For the evaluation of the models, the dataset containing 58 612 reviews is divided into a ratio of 70 : 20 : 10 for training, validation, testing respectively i.e, 41 028 reviews for training, 11 722 for validation, 5862 for testing. For any text classification purpose, the models RNN, GRU LSTM, and Bi-LSTM provide state-of-the-art results as they capture the sequential information over time distributed series data. The comparison of the training, validation, testing accuracies between the proposed model with other state-of-the-art models is listed in Table 4. A plot between training accuracy and validation accuracy per epochs in the 2nd BERT model is shown in Fig. 10. From the plot, it can be clearly noted that both the accuracies have an increasing curve for the last few epochs. This proves the fact that no overfitting of the model occurs during training thus eliminating chances of any decrease in its test accuracy. The use of Dropout layers in the BERT model ensures that overfitting does not occur. Dropouts make use of a probabilistic approach to remove inputs during training and prevent overfitting due to a layer's over-dependence on a few of its inputs.

Fig. 11 shows a plot between the Area Under Curve (AUC) values for different phases of the 1st BERT model.

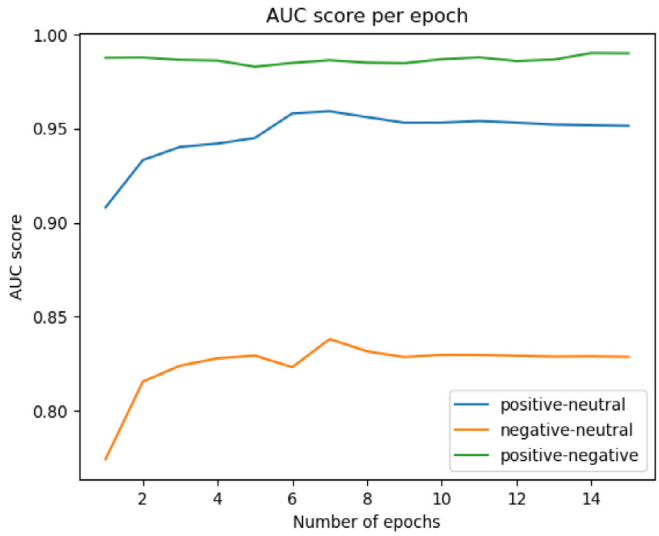

Fig. 11. A plot between the AUC score achieved per epoch for each of the phases for the BERT model 1.

Table 5
Precision, Recall, and F1-Score of our model for the sentiment classification task.

Sentiment	Precision	Recall	F1 score	Support
−1	0.81	0.82	0.82	420
0	0.81	0.66	0.73	785
1	0.95	0.98	0.96	4657
Macro Avg.	**0.86**	**0.82**	**0.84**	**5862**

Table 6
Precision, Recall, and F1-Score of RNN for the sentiment classification task.

Sentiment	Precision	Recall	F1 score	Support
−1	0.58	0.77	0.66	427
0	0.72	0.12	0.20	738
1	0.90	0.99	0.94	4697
Macro Avg.	**0.73**	**0.63**	**0.60**	**5862**

Table 7
Precision, Recall, and F1-Score of GRU for the sentiment classification task.

Sentiment	Precision	Recall	F1 score	Support
−1	0.80	0.63	0.71	446
0	0.63	0.63	0.63	741
1	0.95	0.97	0.96	4675
Macro Avg.	**0.79**	**0.74**	**0.76**	**5862**

From the graph, it can be seen that maximum AUC is shown for the phase which classifies the positive and negative reviews as these sentiments have the maximum amount of reviews for training. Also, the AUC is least for the negative and neutral classification phase as these sentiments have the least number of reviews for training.

The macro average finds the average over the performance for individual classes rather than observations.

The macro-averaged precision for each class can be measured as:

$$Precision_{macro} = \frac{1}{3}\sum_{k=1}^{3}\frac{TP_k}{TP_k + FP_k} = \frac{\sum_{k=1}^{3} Precision_i}{3} \tag{14}$$

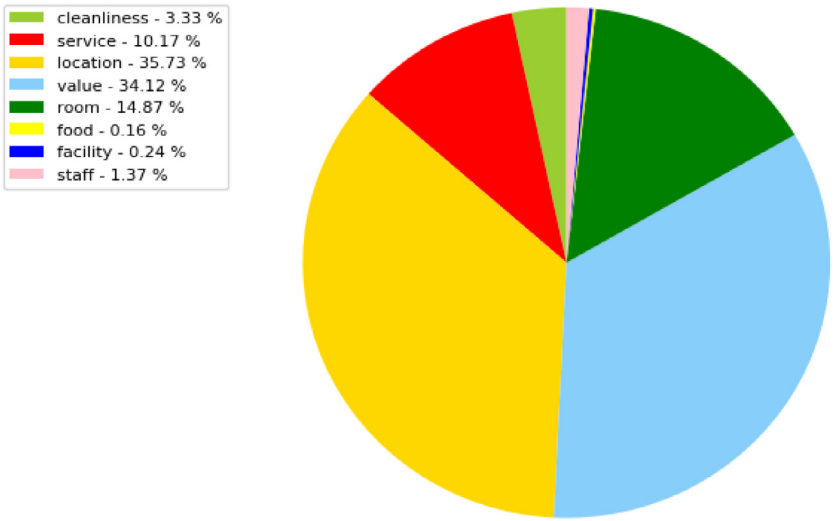

Fig. 12. Pie-chart for category wise hotel review distribution.

Fig. 13. Clustering of aspect categories using K-RMS algorithm on determining words of hotel reviews.

Table 8
Precision, Recall, and F1-Score of LSTM for the sentiment classification task.

Sentiment	Precision	Recall	F1 Score	Support
−1	0.67	0.85	0.75	433
0	0.74	0.26	0.39	703
1	0.92	0.99	0.95	4726
Macro Avg.	**0.78**	**0.70**	**0.70**	**5862**

Table 9
Precision, Recall, and F1-Score of Bi-LSTM for the sentiment classification task.

Sentiment	Precision	Recall	F1 score	Support
−1	0.87	0.56	0.68	448
0	0.63	0.55	0.59	739
1	0.93	0.98	0.96	4675
Macro Avg.	**0.81**	**0.70**	**0.74**	**5862**

Similarly, the macro-averaged recall for each class can be measured as:

$$Recall_{macro} = \frac{1}{3}\sum_{k=1}^{3}\frac{TP_k}{TP_k + FN_k} = \frac{\sum_{k=1}^{3} Recall_i}{3} \tag{15}$$

where, TP = True Positive, FP = False Positive, FN = False Negative

Finally from the macro-averaged recall and precision, the macro F1-score for each class can be measured as:

$$F1_{macro} = 2\frac{Recall_{macro} \cdot Precision_{macro}}{Recall_{macro} + Precision_{macro}} \tag{16}$$

Tables 5, 6, 7, 8, 9 tabulate the recall, precision, and the F1-score for our proposed method, RNN, GRU, LSTM, Bi-LSTM respectively.

From Tables 5, 6, 7, 8, 9 it can be seen that our method produces better results for all of the classes than the state-of-the-art models considered here for comparison. From Table 4 it can be inferred that our method produces training, validation, testing accuracies of 94.35%, 92.79%, and 92.36% which is more than state-of-the-art models. Due to an abundance of reviews falling in the positive polarity (sentiment), the values of different metrics for positive reviews are much greater than of other classes. It is seen that the lowest metrics are for the neutral class due to the sparseness of those reviews in the dataset. But it can also be clearly seen that the values of the metrics for the neutral class is much more than of the state-of-the-art methods. This is because higher weights are allotted to the neutral class during the training of the proposed model.

State-of-the-art models train a standard left-to-right language model and a reverse language model that predict previous words from subsequent words. Thus, in these models, the sentences are processed sequentially — one word per time step. Whereas BERT uses a Transformer instead of the LSTM. It is to be noted that BERT Transformer models are attention-based models. Here, instead of predicting the next word after a sequence of words, BERT randomly masks words in the sentence and predicts them. This enforces the model to learn the usage of information from the complete sentence in deducing the missing words. Eventually, this helps the model to use information from the entire sentence simultaneously regardless of the position of a word in the sentence, thereby helping in better predictions. A Random Forest classifier is used in the ensemble to take advantage of the textual features of the reviews. Also, the training process using a Random Forest becomes more computationally efficient. It is experimentally proven that the Random Forest classifier works better than state-of-the-art neural network models for a dataset with smaller dimension. As the dataset used here for classification contains only 58 612 reviews, Random Forest works much better than other neural network-based models. Moreover, the addition of weights to balance the classes in Random Forest increases its ability to predict the neutral reviews having a lesser amount of data than other state-of-the-art models. Furthermore, the Random Forest model requires less pre-processing, and the training process is much simpler. Owing to these facts, the proposed system can overcome the particular limitations of any single model by combining the outputs of all the models efficiently.

Some of the advantages of the proposed sentiment analysis methodology are (a) the problem of lesser amount of neutral class data is handled by merging two phases of a BERT binary classification model giving higher weights to the neutral reviews, (b) Ensemble of different models has led to a better and more accurate system beyond the reach of any single model, and (c) An advanced TPU v3-8 architecture ensures much faster training.

6.3. Review categorization

The various categories into which the reviews are grouped, and their representative index terms found by manual inspection are shown below:

1. Cleanliness — satisfactory, ample, hygienic, proper, spotless, odor, dirty, clean, smell
2. Service — desk, check in, check out, reliable, fast, convenient, service
3. Location — railway, view, station, airport, distance, far, close, train, metro, transport, market, mall, surrounding, areas, highway, traffic, out
4. Value — price, amount, rate, cheap, worth, low, money, economical, reasonable, fee, expensive, charge
5. Room — bed, bunk-beds, toilet, bathroom, shower, dryer, fridge, space, spacious, outdated, noisy
6. Food — drink, breakfast, spicy, food, tasty, tea, coffee, buffet, bar, restaurant, dinner, lunch, brunch, delicious
7. Facility — front, pool, gym, wifi, spa, internet, wireless, broken, parking, ventilation
8. Staff — friendly, helpful, reliable, quick, good, polite, staff

The distribution of the reviews based on their categories can be visualized by the pie-chart shown in Fig. 12.

From the pie-chart, it can be seen that most of the reviews are related to the location and value categories. The sole purpose of using a clustering approach and including results for word categorization is that we want to check out if the words really clustered around the labeled words and could actually be differentiated visually. There are various clustering algorithms available, but we have used the K-RMS algorithm as this has previously been applied successfully for categorization purposes. Besides, we have included computing time for each clustering algorithm in Table 10, from which it is clear that the K-RMS algorithm takes lesser time compared to other algorithms in functioning. Both K-Means and K-RMS have a time complexity of $\mathbb{O}(n)$ whereas Hierarchical clustering has a time complexity of $\mathbb{O}(n^2)$ and remaining algorithms have a time complexity of $\mathbb{O}(n^3)$ clearly explaining the difference in computing times.

Table 10
Time taken by various clustering algorithms for categorizing reviews into the corresponding categories.

Clustering algorithm	Time taken (s)
K-RMS clustering [41]	8
K-Means clustering [42]	13
Ward Hierarchical clustering [43]	36
Spectral clustering [44]	74
Agglomerative clustering [43]	88

Fig. 14. Word cloud for the determining words of hotel reviews.

The K-RMS algorithm (an optimized version of K-Means) [41] is applied to the determining words i.e., an index set elements for each category. The points represented as clusters can be visualized in the plot in Fig. 13. Each cluster centroid refers to the corresponding aspect category class and every other point assigned to the cluster is the determining word for the class.

From the word cloud for the dataset represented in Fig. 14, it can be seen that most of the commonly recurring words are either the predefined categories or the words present in the index sets of each category.

Due to the presence of some ambiguous words in the index set of each category, a conflict may arise among each of the predefined categories. Such words are: “satisfactory”, “ample”, “proper”, “convenient”, “out”, “low”, “good”, etc. These words can have different meanings based on their respective sentences. Thus, these words can fall in the index set of multiple categories.

7. Conclusion

Helping a user to choose a proper hotel based on his/her requirement and affordability from the online hotel reviews made by the customer gives us an interesting research field called the hotel recommendation system. This ensures that the customers can make optimal travel decisions based on the input query. In this work, we have presented a novel approach for a user query based recommendation system which gives hotels and reviews corresponding to them if required as output as per the user queries. In contrast to the past works that used aspect based Sentiment Analysis, the proposed approach focuses on classifying sentiments of the reviews first, and then group the reviews into different aspect based categories. Finally, based on the user query input, an appropriate hotel along with their reviews is selected. Our experiments are based on a hotel review dataset collected from TripAdvisor.com. For the sentiment classification task, our proposed method produces a classification accuracy 92.36%, a macro average precision of 0.86, recall of 0.82, and F1-score of 0.84 which are much higher than other state-of-the-art models. The results of the categorized reviews form compact clusters using the K-RMS clustering algorithm [41]. Much practical significance may be derived from this study. The proposed system can be incorporated into the tourism systems in various ways. Instead of rigorous searching of hotels with better reviews present in various online websites, our approach can be adopted by the travelers to find hotels within a particular location with particular aspects such as better staff, value, etc. and reviews. Despite achieving good results by our model, there are few limitations which can be taken care of in the future. Firstly, because recent reviews are more regularly browsed, examining how consumer's approval of reviews change over time may provide us additional information. Methods for solving class imbalance such as Random Over-Sampling or Under-Sampling, Bootstrap Aggregating (Bagging) based techniques, or Boosting-based techniques could be tested to achieve better results. Methods such as the Fuzzy ensemble method or Dempster–Shafer method, etc. could be experimented with to form an ensemble of the methods. The categorization results could only be checked manually which could be prone to human-level errors. The availability of more qualified annotators may help in reducing such human-level errors. Besides, if proper multilingual datasets are found, our method can then be used on such datasets just by changing the vector encoding mode.

CRediT authorship contribution statement

Biswarup Ray: Conceptualization, Data curation, Formal analysis, Resources, Software, Methodology, Writing - original draft. **Avishek Garain:** Conceptualization, Data curation, Formal analysis, Resources, Software, Methodology, Writing - original draft. **Ram Sarkar:** Methodology, Supervision, Writing - original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Annex

Extracting Subjectivity score from a review:

```
from textblob import TextBlob

blob = TextBlob("The rooms are very spacious and well maintained.")
print(blob.sentiment)
'''
Output:
Sentiment(polarity=0.8, subjectivity=0.75)
'''
```

Data cleaning operations on the crawled reviews:

```
import re
import string
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

def POS_tags(review):
    text = nltk.word_tokenize(review)
    return nltk.pos_tag(text)

def lemmatize(word):
    lemmatizer = WordNetLemmatizer()
    return lemmatizer.lemmatize(word)

def contract_whitespace(review):
    review = review.replace('\n', ' ')
    review = re.sub("\s\s+", " ", review.strip())
    return review

def remove_stopwords(sentences):
    for i in range(len(sentences)):
    words = nltk.word_tokenize(sentences[i])
    words = [word for word in words if word not in stopwords.words('english')]
    sentences[i] = ' '.join(words)
    return "".join(sentences)

#Functions related to Pascal Scaling
def partitioner(hashtag,words):
    while hashtag:
    word_found=longest_word(hashtag,words)
    yield word_found
    hashtag=hashtag[len(word_found):]

def longest_word(phrase,words):
    current_try=phrase
    while current_try:
        if current_try in words or current_try.lower() in words:
            return current_try
        current_try=current_try[:-1]
    return phrase

def split_uppercase(s):
    r = []
    l = False
    for c in s:
        # l being: last character was not uppercase
        if l and c.isupper():
            r.append(' ')
        l = not c.isupper()
        r.append(c)
    return ''.join(r)

def partition_hashtag(text, words):
    return re.sub(r'#(\w+)',lambda m:''.join(partitioner(m.group(1),words)),text)

def read_dictionary_file(tweet):
    return set(word.strip() for word in tweet)
```

Methods for additional feature extraction:

```
import nltk
import json
from sklearn.feature_extraction.text import
                          TfidfVectorizer


def freqeunt_terms_extraction_unigram_bigram
                  (texts, ngram_range=(1,2),
    n_terms=None):
    """
    Extract frequent terms using simple TFIDF
              ranking in given list of texts
    """
    tfidf_model = TfidfVectorizer(lowercase=True,
           ngram_range=ngram_range, stop_words=None,
           min_df=5, max_df=0.8)
    uni_model=tfidf_model.fit(texts)
    X = tfidf_model.fit_transform(texts)
    vocabulary_sort = [v[0] for v in sorted
            (tfidf_model.vocabulary_.items(),
                        key=operator.itemgetter(1))]
    ranks = np.array(np.argsort(X.sum(axis=0))).ravel()
    frequent_terms = [vocabulary_sort[r] for r in ranks]
    frequent_terms = [f for f in frequent_terms
                                if len(f) > 3]
    return uni_model,ranks,frequent_terms
```

References

[1] M. Day, C. Lee, Deep learning for financial sentiment analysis on finance news providers, in: 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2016, pp. 1127–1134.

[2] A. Devitt, K. Ahmad, Sentiment polarity identification in financial news: A cohesion-based approach, 2007.

[3] X. Li, H. Xie, L. Chen, J. Wang, X. Deng, News impact on stock price return via sentiment analysis, Knowl.-Based Syst. 69 (2014) http://dx.doi.org/10.1016/j.knosys.2014.04.022.

[4] J.Z.G. Hiew, X. Huang, H. Mou, D. Li, Q. Wu, Y. Xu, Bert-based financial sentiment index and LSTM-based stock return predictability, 2019, arXiv:1906.09024.

[5] L. Zhao, L. Li, X. Zheng, A bert based sentiment analysis and key entity detection approach for online financial texts, 2020, arXiv:2001.05326.

[6] M. Bhat, M. Qadri, M. Kundroo, N. Ahanger, B. Agarwal, Sentiment analysis of social media response on the covid19 outbreak, Brain Behav. Immun. (2020) http://dx.doi.org/10.1016/j.bbi.2020.05.006.

[7] K. Manguri, R. Ramadhan, P. Mohammed Amin, Twitter sentiment analysis on worldwide COVID-19 outbreaks, Kurdistan J. Appl. Res. (2020) 54–65, http://dx.doi.org/10.24017/covid.8.

[8] G. Ruz, P.A. Henríquez, A. Mascareño, Sentiment analysis of Twitter data during critical events through Bayesian networks classifiers, Future Gener. Comput. Syst. 106 (2020) http://dx.doi.org/10.1016/j.future.2020.01.005.

[9] R.K. Amplayo, M. Song, An adaptable fine-grained sentiment analysis for summarization of multiple short online reviews, Data Knowl. Eng. 110 (2017) http://dx.doi.org/10.1016/j.datak.2017.03.009.

[10] A. Abdi, S.M. Shamsuddin, S. Hasan, Machine learning-based multi-documents sentiment-oriented summarization using linguistic treatment, Expert Syst. Appl. 109 (2018) http://dx.doi.org/10.1016/j.eswa.2018.05.010.

[11] D. Ghosh, A sentiment-based hotel review summarization, 2020, pp. 39–44, http://dx.doi.org/10.1007/978-981-13-7403-6_5.

[12] L. Mostafa, Machine learning-based sentiment analysis for analyzing the travelers reviews on Egyptian hotels, 2020, pp. 405–413, http://dx.doi.org/10.1007/978-3-030-44289-7_38.

[13] W. Kasper, M. Vela, Sentiment analysis for hotel reviews, 2011.

[14] L. Dey, S. Chakraborty, A. Biswas, B. Bose, S. Tiwari, Sentiment analysis of review datasets using Naïve Bayes' and K-nn classifier, Int. J. Inform. Eng. Electron. Bus. 8 (2016) 54–62, http://dx.doi.org/10.5815/ijieeb.2016.04.07.

[15] N. Akhtar, N. Zubair, A. Kumar, T. Ahmad, Aspect based sentiment oriented summarization of hotel reviews, Procedia Comput. Sci. 115 (2017) 563–571, http://dx.doi.org/10.1016/j.procs.2017.09.115.

[16] C.-F. Tsai, K. Chen, Y.-H. Hu, W.-K. Chen, Improving text summarization of online hotel reviews with review helpfulness and sentiment, Tour. Manag. 80 (2020) 104122, http://dx.doi.org/10.1016/j.tourman.2020.104122.

[17] E. Marrese-Taylor, J. Velasquez, F. Bravo-Marquez, A novel deterministic approach for aspect-based opinion mining in tourism products reviews, Expert Syst. Appl. 41 (2014) 7764–7775, http://dx.doi.org/10.1016/j.eswa.2014.05.045.

[18] S. Palakvangsa-Na-Ayudhya, V. Sriarunrungreung, P. Thongprasan, S. Porcharoen, Nebular: A sentiment classification system for the tourism business, 2011, http://dx.doi.org/10.1109/JCSSE.2011.5930137,

[19] J. Carrillo-de Albornoz, L. Plaza, P. Gervás, A. Díaz, A joint model of feature mining and sentiment analysis for product review rating, 2011, pp. 55–66, http://dx.doi.org/10.1007/978-3-642-20161-5_8,

[20] Y.-H. Hu, Y.-L. Chen, H.-L. Chou, Opinion mining from online hotel reviews – a text summarization approach, Inf. Process. Manage. 53 (2017) 436–449, http://dx.doi.org/10.1016/j.ipm.2016.12.002.

[21] P.-J. Lee, Y.-H. Hu, K.-T. Lu, Assessing the helpfulness of online hotel reviews: A classification-based approach, Telemat. Inform. 35 (2018) http://dx.doi.org/10.1016/j.tele.2018.01.001.

[22] M.P. O'Mahony, B. Smyth, A classification-based review recommender, Knowl. Based Syst. 23 (2010) 323–329.

[23] Y.-H. Hu, K. Chen, Predicting hotel review helpfulness: The impact of review visibility, and interaction between hotel stars and review ratings, Int. J. Inf. Manage. 36 (2016) 929–944, http://dx.doi.org/10.1016/j.ijinfomgt.2016.06.003.

[24] N. Hu, I. Bose, N. Koh, L. Liu, Manipulation of online reviews: An analysis of ratings, readability, and sentiments, Decis. Support Syst. 52 (2012) 674–684, http://dx.doi.org/10.1016/j.dss.2011.11.002.

[25] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv:1810.04805.

[26] Tensor processing units (TPUs) documentation, URL: https://www.kaggle.com/docs/tpu.

[27] A. Garain, Hotel reviews from around the world with sentiment values and review ratings in different categories for natural language processing, 2020, http://dx.doi.org/10.21227/8ggw-hm23.

[28] A. Garain, S.K. Mahata, S. Dutta, Normalization of numeronyms using NLP techniques, in: 2020 IEEE Calcutta Conference (CALCON), IEEE, 2020, pp. 7–9.

[29] A. Garain, A. Basu, The titans at semeval-2019 task 5: Detection of hate speech against immigrants and women in Twitter, in: Proceedings of the 13th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Minneapolis, Minnesota, USA, 2019, pp. 494–497, http://dx.doi.org/10.18653/v1/S19-2088, URL: https://www.aclweb.org/anthology/S19-2088.

[30] A. Garain, S.K. Mahata, Sentiment analysis at sepln (tass)-2019: Sentiment analysis at tweet level using deep learning, 2019.

[31] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, 2016, arXiv preprint arXiv:1609.08144.

[32] XLA: optimizing compiler for machine learning: TensorFlow, URL: https://www.tensorflow.org/xla.

[33] J. Zhan, H.T. Loh, Y. Liu, Gather customer concerns from online product reviews - a text summarization approach, Expert Syst. Appl. 36 (2009) 2107–2115.

[34] R. Ř.ehůřek, P. Sojka, Software framework for topic modelling with large corpora, in: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, ELRA, Valletta, Malta, 2010, pp. 45–50, http://is.muni.cz/publication/884893/en.

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[36] fuzzywuzzy, URL: https://pypi.org/project/fuzzywuzzy/.

[37] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536, http://dx.doi.org/10.1038/323533a0.

[38] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, 2014, CoRR, abs/1406.1078, URL: http://arxiv.org/abs/1406.1078.

[39] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[40] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, IEEE Trans. Signal Process. 45 (11) (1997) 2673–2681.

[41] A. Garain, D. Das, K-RMS algorithm, Procedia Comput. Sci. 167 (2020) 113–120.
[42] J.A. Hartigan, M.A. Wong, Algorithm AS 136: A k-means clustering algorithm, J. R. Statist. Soc. Ser. C 28 (1) (1979) 100–108.
[43] S.C. Johnson, Hierarchical clustering schemes, Psychometrika 32 (3) (1967) 241–254.
[44] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: Advances in Neural Information Processing Systems, 2002, pp. 849–856.