A Deep Learning-based Sentiment Analysis Approach for Online Product Ranking with Probabilistic Linguistic Term Sets

Zixu Liu, Huchang Liao, *Senior Member, IEEE*, Maolin Li, Qian Yang, Fanlin Meng

***Abstract*—The Probabilities linguistic term set (PLTS) is an efficient tool to represent sentimental intensities hidden in unstructured text reviews which are useful for multi-criteria online product ranking. Traditional machine learning-based sentiment analysis methods adopted in existing studies to obtain PLTSs often result in unsatisfying prediction accuracy and thus inevitably affect product ranking results. To overcome this limitation, in this study, we propose a deep learning-based sentiment analysis approach to produce PLTSs from online product reviews to rank online products. A natural language processing-based method is first applied to extract product features and corresponding feature texts from online reviews. Then, state-of-the-art deep learning-based models are implemented to conduct the sentiment classification for online product/feature review texts. To ensure classification accuracy, we propose an experimental matching mechanism to identify the level of sentiment tendency for all rating labels of a review dataset and then match each label with the most appropriate linguistic term. The experiment results reveal: 1) our matching mechanism can benefit the training of a text classification model to identify sentiment tendencies from review texts with high prediction accuracy; 2) with the help of the trained classification model, our approach can predict sentimental intensities of the extracted features' texts in the form of PLTSs with competitive accuracy. A case study of applying PLTSs output from our approach to an online product decision-making problem is also provided to validate the applicability of our approach.**

***Index Terms*— Sentiment analysis, Text reviews, Text classification, Deep learning, Probabilistic linguistic term set.**

I. Introduction

ONLINE review systems in current e-commerce platforms become an important information source to affect customers' decisions regarding online shopping [1]. To support customers with extracting useful information from large amounts of reviews and making a purchase decision from a set of products/services regarding multiple criteria, the accurate measurement of online reviews is worth investigating [2], [3]. Probabilities linguistic term set (PLTS) [4], which combines linguistic terms with probabilities to enhance the flexibility and comprehensiveness of uncertain information expression, is an efficient tool to represent sentimental intensities hidden in unstructured text reviews. PLTS has been widely applied to represent linguistic evaluations for text online reviews in multi-criteria online product ranking problems under uncertainty [2], [3], [5]–[8].

A. Challenges and Research Questions

In the current research [1], [9], the common way to address the problem of product ranking based on online reviews is composed of three stages: 1) product features extraction from online reviews, 2) sentiment analysis for calculating the overall sentiment scores of sentiment words of review texts, 3) ranking alternative products based on the results of the first two stages.

A.1 Research of Feature Extraction

Apart from the overall comment for a product, online reviews always contain descriptions and preferences for different product features that will affect the purchasing behavior of a customer. Thus, product features and corresponding sentiment tendencies need to be considered when ranking products. How to effectively extract product features from a large set of online reviews is the basis and an important step of the online review analysis problem [10]. The most widely used method in current research to extract production features is the statistical-based method. The Latent Dirichlet Allocation (LDA) model is a typically generative statistical model. For instance, Tirunillai and Tellis [11] used the LDA to extract the key latent dimensions of consumer satisfaction with quality. Guo et al. [12] and Bi et al. [13] used LDA to extract the features of products/services from online reviews to identify the preferences of customers. The LDA model could identify a number of topics from text documents, where each topic contains several words that are representative of that topic. In other words, the output of the LDA model is the sparse representation of a text, it only keeps the key features which are not related to each other and ignores the irrelevant

Manuscript received November 14, 2022; revised April 04, 2023; accepted April 25, 2023. This work was supported by the National Natural Science Foundation of China under Grant 71971145 and 72171158. *(Corresponding Author: Huchang Liao)*.

Z.X. Liu is with the Southampton Business School, University of Southampton, Southampton, SO17 1BJ, UK (e-mail: Zixu.Liu@soton.ac.uk).

H. C. Liao and Q. Yang are with the Business School, Sichuan University, Chengdu 610064, China (e-mail: liaohuchang@163.com; yangqianscu@163.com).

M.L. Li and F.L. Meng are with the Department of Computer Science and Alliance Manchester Business School, respectively, The University of Manchester, Manchester, M13 9PL, UK (e-mail: limaomao.maolin@gmail.com; fanlin.meng@manchester.ac.uk).

information. Consequently, the LDA model cannot be applied or extended to retrieve sentiment phrases or sub-sentences that describe a particular feature in a straightforward manner.

Other statistical-based methods for feature extraction include association rule mining [14], hidden Markov model [15], and conditional random fields [16]. However, the results of these kinds of methods heavily depend on the training corpus, which is often time-consuming, expensive, and difficult to construct. The above-mentioned studies chose to avoid building a new corpus but use existing public corpora to train their tools. However, the source of raw texts in most of the corpora is news, which is quite different from online reviews of products [1]. Such a limitation lowers the accuracy of the product feature extraction and further sentiment analysis results, which limits the utilization of statistical-based methods in online review analysis. By identifying this research gap, our first research objective is to develop an approach to extract product features and corresponding sentiment phrases or sentences efficiently and accurately from online reviews, and then analyze the sentiment tendencies of the reviews from each product concerning each product feature.

A.2 Sentiment Classification on Review Texts

In the second stage, sentiment analysis methods for the review texts and extracted features' corresponding texts or sub-sentences are needed. The commonly used sentiment analysis methods to calculate the sentiment score of each product feature and overall product satisfaction from the review text are based on machine learning (ML). The basic idea of these methods is to treat the sentiment analysis of online reviews as a text classification problem [1]. In ML, the text classification aims to determine the category (or label, class) of a given text, and the classification result can be a binary classification or multi-class classification results [17]. Note that the probabilistic distribution on all possible labels or classes from the classification result matches the mathematical form of PLTSs perfectly. Many studies [2], [3], [5] used the rule-based sentiment analysis tool in a software package called Stanford CoreNLP to produce PLTSs from online reviews, with classified labels and the corresponding posterior probability of each label. Different from ML methods, the rule-based sentiment analysis methods do not require additional annotator efforts to collect new training data. The rule-based sentiment analysis tool in the Stanford CoreNLP is an already well-trained compositional model over trees using deep learning (DL) [18], which enables to get linguistic annotations quickly and painlessly from a text. The tool provides conveniences for users to use by calling their application programming interface. However, its flexibility and scalability are insufficient for producing PLTSs since it was not designed specifically for PLTS generation but to provide a general sentiment analysis toolkit with a lightweight framework. To implement the rule-based sentiment analysis method, the sentiment toolkit was pre-trained on a particular dataset which is different from online product reviews regarding customer preferences. Thus, sentiment analysis results from this tool are not quite accurate, which inevitably affects the final stage of product ranking. Although re-pre-training the sentiment toolkit is possible by using a new dataset, the construction of such a corpus, especially annotating the sentiment scores for all the text spans or product features in a text, requires extensive manual efforts which can be expensive and time-consuming.

As we discussed before, the production of PLTSs from online text reviews can be considered as a sentiment classification problem. Supervised ML-based sentiment analysis methods are the primary choices by current research since they can enrich the semantic expression. The biggest advantage of such kind of method is that different supervised ML methods can be used for different problems of sentiment classification (i.e., categories of ranking products in different problems are different), which improves the accuracy of classification results [1]. Naive Bayes is a probability-based classification method used in some studies [19], [20] to do sentiment analysis for online reviews. This classifier can determine the sentiment category/class of a text according to the joint probability of text feature items and sentiment categories. The main drawback of this method lies in the assumption that all features in the text are independent, which limits the classification performance because usually, the words in a text are not totally independent. Support vector machine (SVM) is another supervised ML algorithm used in many studies for sentiment classification [21]–[24]. The SVM classifier works by deciding a classifying hyperplane where data points are above or below it. In other words, there is no probabilistic explanation for the classification. Therefore, only sentiment orientations of online reviews are predicted, and PLTSs cannot be obtained by this classifier. Furthermore, the SVM algorithm is not suitable for large data sets, especially for online review text data, since their target classes are inevitably overlapping which makes it harder for the SVM to predict the hyperplane for classification. Decision tree-based ensemble algorithms, such as random forest and gradient boosting decision tree algorithms, were also discussed and applied in a few studies [9], [25], [26] to solve the problem of sentiment classification. Onan et al. [27], [28] conducted a comprehensive analysis to show that Decision tree-based ensemble algorithms could get a higher classification accuracy compared with base-learning algorithms (SVM, Naive Bayes) in text classification. However, such kind of algorithms are not suitable for tabular data, such as image and text data with the same meaning in all properties or dimensions [29]. From the above-mentioned literature, we find that the lack of the training dataset is the main barrier to implementing a general supervised ML-based sentiment classification method for online reviews. More specifically, as Onan and Korukoglu mentioned in [30], the lack of an abundant amount of training data makes it difficult to train ML-based sentiment classification algorithms in a feasible time and degrades the classification accuracy of the built model. The reason for lack of the data is that the dataset of labelled training samples with sentiment orientations for a kind of products is usually hard to find, since only the reviews of products in the exact same product category can be used for training [1]. Furthermore, the supervised ML-based methods cannot generate a convinced classification accuracy for the sentiment

classification problem because they cannot efficiently process a large text dataset as discussed before.

Aspect-Based Sentiment Analysis (ABSA) is a sentiment classification method which has been researched in many studies [31]–[33]. It receives a set of texts such as product reviews discussing a particular product as input, then attempts to detect the main or the most frequently discussed aspects (features) of the product (e.g., 'screen') and estimate the average sentiment of the texts per aspect. However, it has the following drawbacks. First, product reviews normally contain a few short sentences, therefore some important features from the other aspects may be omitted by ABSA if a review text mentions several aspects of the product (which always happens in product reviews). Second, if combining all reviews as a single document to input to ABSA, only an average sentimental score of all reviews for each identified feature is classified and output. In this scenario, the output cannot be used to do the third stage of product ranking mentioned before, as the sentiment score of a feature for each mentioned review text is required in this stage. Solely classic text classification ML methods were applied in most ABSA models [32], [33] which cannot perform the sentiment classification-related tasks well.

A.3 Implementation of DL Models on Text Classification

The implementation of DL is growing fast in recent years for supervised learning tasks. DL has been proven to improve the defects such as data sparsity, dimension explosion, and poor generalization ability of traditional ML methods on text classification [34], [35]. Classifiers based on DL avoid the cumbersome pre-processing process and have strong learning abilities and higher prediction accuracy [17]. Efficiently managing emerging technologies, especially DL in artificial intelligence, could bring significant opportunities by automatically identifying customers' needs to enable innovation, profit and growth [36], [37]. Lu et al. [38] proposed a patent citation classification model which integrated state-of-art DL models. The experimental results show that the model effect based on deep learning is significantly better than the traditional text representation and classification method. Based on state-of-art DL text classification models, Jiang et al. [39] proposed a model TechDoc for the accurate automated classification of technical documents. Wang et al. build a CNN-based text classification model to map customers' needs to production specifications by sentiment analyzing customers' product reviews [40]. As far as we know, studies of implementing DL models for generating PLTSs have not been well-studied. Except for mentioned studies, only a few studies in ABSA [32], [33] and sentiment analysis on text reviews [41]–[43] employed general DL models such as the Deep Neural Network and convolutional neural network (CNN) but still neglected the state-of-art DL models. In addition, we also notice that most of the existing studies on sentiment analysis with PLTSs lack of large training datasets. Therefore, one cannot explore the capacity of DL models in generating PLTSs for online product ranking problems. Besides, most online shopping websites ask their customers to rate products in the range of 1-5 stars. How to match these rating labels with correct linguistic terms is still an open problem. Due to individualized cognition, the same word may mean differently to different people, especially for sentiment words in text reviews that are posted by different reviewers with different linguistic and cultural backgrounds [2], [44]. As the example shown in [2], two reviewers both use the same sentiment word "great" for a TV but provide different rating scores. In this case, "great" approximately means 0.75 in the cognition of the reviewer with a lower score but implies 1 for the reviewer with a higher score. The phenomenon indicates that the rating score represents the sentiment tendencies and only roughly coarse-grained sentiment intensities of the review text but cannot precisely reflect fine-grained sentiment intensities in the review text. Therefore, directly using the 1-5 rating scores as the supervised labelling information of review texts in the training process is inappropriate and cannot get a text classification model with high prediction accuracy. Since the PLTSs represent sentimental intensities hidden in unstructured text reviews, we conclude that directly matching 1-5 rating labels with 5 linguistic terms *{"very negative", "negative", "neutral", "positive", "very positive"}* is inappropriate that cannot help to train a model to generate the correct PLTSs from text reviews. The field of sentiment analysis generally aims to classify texts as *negative*, *neutral* and *positive* [45]. But the existing work only divided reviews as "positive" or "negative" to match with linguistic terms, which was not accurate enough and only provided a rough sentiment tendency [19], [21], [22]. Hence, identifying the level of the sentiment intensities of the review text and matching it with the LST *{"negative", "neutral", "positive"}* is more reasonable and could generate more accurate PLTSs. Identifying the above research gaps, we will explore our second, third, and fourth research problems: how to obtain a large enough dataset consisting of labelled training samples with sentiment orientations for different products under different categories? How to apply state-of-the-art DL models to solve the problem of sentiment classification and obtain PLSTs from review texts for the product and its features? Which model has the best performance in what kind of case?

B. Motivation and Contributions

To solve the above-mentioned research problems, this paper presents a DL-based sentiment analysis approach to produce PLSTs from online reviews to rank online products. The state-of-the-art DL-based text classification models are implemented as a backbone of the proposed approach to solve the sentiment classification problem. We then map the classification results to the PLSTs which can be directly used in the product ranking. A comparative analysis of reviewed literature (Table XII) is provided in Section VI to demonstrate the advantages of our proposed approach and help the reader to better understand our contributions. We summarize the main contributions of this study as follows:

1) We provide a general algorithm which can be easily generalized to various PLTS scenarios to exact and collect products' information and corresponding review texts from a data source (e.g., a public data source or private dataset). It can output the training dataset that includes labelled

training text samples with sentiment orientations for different products under different product categories.

2) We propose a natural language processing (NLP)-based method to extract product features from online reviews and retrieve the corresponding text or sub-sentences that only describe a particular product feature.
3) We experiment and prove the way of matching rating labels with linguistic terms directly cannot provide accurate PLTSs in identifying the correct sentiment tendencies from online reviews. The existing works [19], [21], [22] simply divided reviews as "positive" or "negative" to match with linguistic terms was not accurate enough and only provided a rough sentiment tendency. An experimental solution to this problem is proposed in our work which is to match the rating scores 1-5 with the LST *{"negative", "neutral", "positive"}* to generate more accurate PLTSs.
4) Several different state-of-the-art DL and ML-based text classification models are implemented and tested in our framework to generate PLSTs for the product and its features from input review texts. Experiment results demonstrate the high prediction accuracy and competitive performance of our approach with DL models in sentiment classification and extracting PLTSs from online reviews.

C. Organization of the Paper

The rest of the paper is organized as follows: In Section II, several state-of-the-art DL-based classification models are reviewed. The detail of our proposed approach is depicted in Section III. In Section IV, we present and analyze the results of our designed experiments. A case study is presented in Section V to illustrate how to apply the PLSTs output by our approach to the problem of product ranking based on online reviews. Section VI discusses the management implication of our work and concludes the paper.

II. Short Review on PLTS and Deep Learning Models

To facilitate further presentation, this section reviews the concept of PLTSs used in online product ranking and several state-of-art text classification models based on DL methods which will be used in this study.

A. Definition of PLTSs

Here we provide the formal definition of PLTS first.

A set of possible values, $S = \{s_0, s_1, \dots, s_q\}$, of a linguistic variable is called a linguistic term set (LTS), where s_α is a linguistic term in $S, \alpha \in \{0,1, \dots, q\}$ and $s_\alpha > s_\beta$ if $\alpha > \beta$. To describe the complex linguistic evaluation information with hesitancy or uncertainty given by experts, Pang et al. [4] proposed the concept of PLTS as $h_S(p) = \{s_\alpha(p_\alpha) | \alpha = \{0,1, \dots, q\}\}$ by associating each linguistic term in S with a probability. Later, Wu and Liao [2] enhanced the representation of the PLTS to the form expressed as Eq. (1):

$$h_S(p) = \{s_0(p_0), s_1(p_1), \dots, s_q(p_q), s_s(p_s)\} \quad (1)$$

In Eq. (1), p_α ($p_\alpha \geq 0$) is the probability of the linguistic term s_α, s_s refers to the universal set of S, and p_s is the ignorance part of the probability such that $\sum_{\alpha=0}^{q} p_\alpha + p_s = 1$. Considering the ignorance part of probability is important for deriving a correct decision-making result under uncertainty [2].

As an emerging tool of complex information representation, PLTSs have been applied to represent the evaluation information extracted from online reviews [2], [3], [5]–[8]. These studies can be divided into two categories: one relied on specific technologies (e.g., the cloud model) to summarize numerical online ratings into PLTSs [6]; the other kind of methods depended on different techniques of attribute extraction and sentiment analysis to transform sentiment tendencies and intensities contained in text reviews into PLTSs [2], [3], [5], [7], [8].

B. Deep Learning Models in Text Classification

In this section, several state-of-art text classification models based on DL methods are reviewed.

CNN is a specialized kind of neural network that employs a mathematical operation called convolution. The convolution is used in place of general matrix multiplication and in at least one of the layers in the CNN [35]. TextCNN [46] used CNN to extract key information similar to n-gram in sentences. All parts of the model and their functionalities include: 1) input layer, which inputs the text data into the model, 2) embedding layer, which extracts the text feature representation, 3) convolution layer, which is built by filters of different size and transfers the input data to a feature map, 4) max-pooling layer, which reduces the dimension of the data from the convolution layer, 5) fully connected SoftMax layer, which contains few full connected layers that comply the data extracted by previous layers to form the final output, i.e., the probability of each category/class in binary or multi-class.

Different from the TextCNN which segments the input text data into words, the CharCNN proposed by Zhang et al. [47] accepts a sequence of encoded characters as input. It is the character-level CNN that aims to learn language representation directly from characters. The character encoding in CharCNN was achieved by predefining an alphabet of size *m* which is the number of unique characters in a language, and then randomly initializing an $m \times d$ look-up table matrix, where d is the dimension size of a character vector representation. This look-up table matrix will be updated during training. According to the look-up table, the input of a sequence of characters was then encoded to an $l_0 \times d$ matrix, where l_0 is the maximum length of the character sequence and the characters exceeding this length was ignored.

One problem of CNN is regarding the fixed size of the filter. It is hard to capture the global information since the fixed size of the filter can only model the local information. Focusing on this, Liu et al. [48] developed the TextRNN or bi-directional long short-term memory (bi-LSTM) to capture bi-directional information with the variable length for text classification problems. This model can capture richer context information than conventional LSTM. This is achieved by concatenating the outputs of two individual LSTM units in the architecture of TextRNN where one LSTM processes the sequence from left to

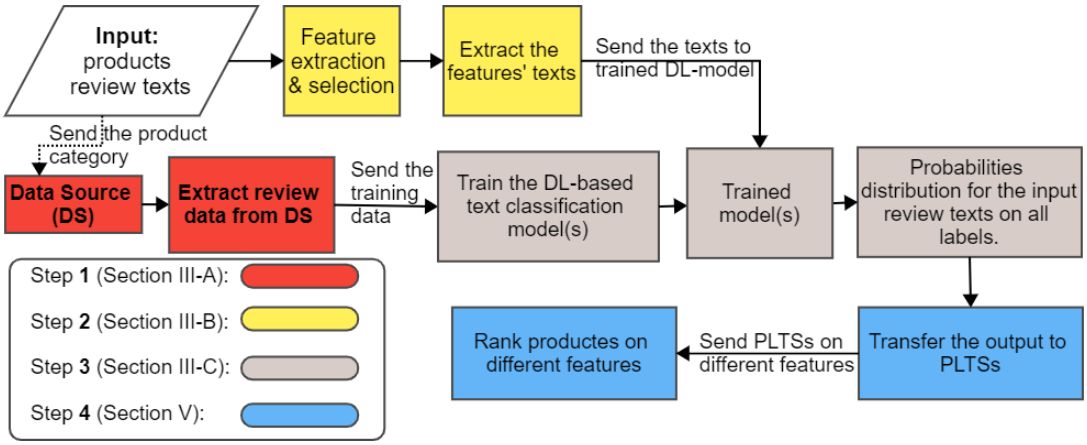

Fig. I. The diagram of the proposed DL-based sentiment analysis approach to generate PLTSs from online reviews

right, and the other from right to left.

Sequence to Sequence (Seq2Seq) is another DL-based classification model within an Encoder-Decoder framework [49]. In the Encoder-Decoder network, the whole source of sentences is compressed by the encoder (e.g., an LSTM) into a single vector and the decoder (e.g., another LSTM) extracts the relevant information from the encoder output and generates a new sequence. However, different parts of the source sentence can be more useful than others. To solve the problem of the long-term dependence on text, the attention mechanism [50] was added to the Seq2Seq model. At each decoding step, this mechanism decides which source parts are important, which allows the model to learn to "focus" on the most relevant source parts for each step. This mechanism can also benefit the Seq2Seq-based classification performance [51].

Like the Seq2Seq, Transformer [52] designed by Google Brain is also based on an encoder-decoder architecture. The main advantage of the Google transformer is to adopt the self-attention mechanism that weights the significance of each part of the input data differentially. Although this model was originally proposed for the task of machine translation, with the development of pre-trained systems such as BERT (Bidirectional Encoder Representations from Transformers) [53], the transformer becomes the most popular architecture for various NLP tasks (e.g., text classification), having generated state-of-the-art results [53], [54]. In [53], it is shown that BERT can be fine-tuned for specific NLP tasks after pre-training with large datasets of English Wikipedia and BooksCorpus [55].

FastText, created by Facebook's AI research lab [56], is a DL-based classification model for efficient learning of word embedding and text classification. The main principle behind this model is that the morphological structure of a word carries important information about the meaning of the word. Such a structure is not considered by traditional word embedding. Despite its lightweight baseline, Joulin et al. [56] showed that fastText has the competitive result in terms of accuracy for training and evaluation, but magnitude faster compared with other DL classifiers.

All the above-mentioned models will be implemented and experimented with in our proposed approach. Transformers, fastText and Seq2Seq are popular sentiment analysis models and have been used in many of current research [57]–[59], which is also why we choose them. The relatively "old" NN-based models -TextCNN, TextRNN and CharNN- which are used in some works of ABBS [32], [33] are chosen to run the comparative experiment.

III. A Novel Sentiment Analysis Approach with PLTSs

In this section, we depict our proposed DL-based sentiment analysis approach to generate PLTSs from online product reviews. Fig. I shows the diagram of the approach. The input of the proposed approach is the review texts of the products that need to be ranked. Before training a sentiment classification model, we need to collect the training data from a data source. The training data is the review texts of all products that belong to the same category as the product of the input reviews. Then, state-of-the-art DL text classification models are selected and used as the backbone model in our approach. Production feature extraction is also an important part of our approach. Compared with the sentimental distribution of the overall review text, the sentiment distribution of the feature contained by the text is more commonly used in ranking products because different features usually have different importance and weight. The input review texts that contain a selected feature from the experts are extracted and retrieve the corresponding sentences that only represent this selected feature. These sentences and input review texts are then fed into the trained model to predict the probabilities distribution among different levels of sentiment tendency. The output can be represented as the PLTSs that be used in the latter stage of product ranking. In the following, Section A presents the part of data collection in our approach, Section B discuss production feature extraction, and Section C illustrates text classification.

A. Data Collection

In this subsection, we describe the method of collecting the training data for different categories of products by adapting existing online product reviews from a public data source. This method could serve as a reference method to help people create a customized review text dataset.

With the fast development of ML and DL, many advanced and sophisticated models have been built for the NLP. The *bias* of the training result normally can be controlled in an acceptable range under these complex models. That is because these

models have a robust text features extraction process to transfer the input texts to the machine-readable format, normally vector representation. Therefore, the *variance* of the training results is one of the main factors that need to be considered. Generally, there are two commonly used methods to deal with the large variance and tackle overfitting: regularization that is to introduce additional terms in the training loss function to penalize extreme parameter values or adding more data [60]. The first one has already been implemented in almost all sophisticated ML models. Hence, applying the second method-collecting more data to get a better training result is the first problem we must solve in our approach.

In this study, we take the use of reviews on the Amazon platform as an example to demonstrate how to use such large data to benefit the research of PLTS generation. The data we used was taken from Ni et al. [61] that included reviews in the range of May 1996 – Oct 2018. This data source contains 233.1 million reviews (34GB) and 15.5 million products (24GB).[1] Two types of files in the dataset are critical to build our own dataset for our approach: complete review data and products metadata. The data format used here is one-review (product)-per-line in JSON[2] in which the data consists of attribute-value pairs and arrays that make the data human-readable. Fig. A.I in the Appendix file Section A shows the data structure of the review data and product metadata. From the product metadata, we use the content of 'category' to match our search query to get the product's '*asin'*. '*asin*' is the Amazon standard identification number. Then, based on '*asin'*, we could get all reviews for this product, and the contents of '*overall'* and '*reviewText'* are extracted to create our training samples. For instance, we can get the following training sample from the sample review in Fig. A.I: 'Text': '*Love this TV and great buy*'; 'lable':*5*.

Algorithm I describes the method to extract a training dataset based on the category information of a product. The category information of a product can be checked on Amazon.com. If this product is not available on the website, searching for a similar product could also obtain the category information. The extra filter condition can be added to Step *4.1* in Algorithm I to filter the unwanted data (e.g., only return the reviews published after 2010) and create a customized training dataset.

Algorithm I: training review data extraction

Input: The *main category* and list of *categories* of product(s)
Output: Review data of all products that have the same *categories* information.
1. Load the JSON file of product metadata for the *main category* as list *JP;* create an empty list *P*.
2. Parallel read all the contents (products' metadata) in *JP*.
   2.1. If the value of the '*category'* of this product equals the input list of *categories:*
      2.1.1. Append the values of '*asin'* to the list *P*.
3. Load the JSON file of review data for the *main category* as list *RJ;* create an empty list *R*.
4. Parallel read all the contents (review data) in *RJ*.
   4.1. If the value of the '*asin'* of this review data in the list *P:*
      4.1.1. Append an array that contains the values of '*revewText'* and '*overall'* to the list *R*.
5. Save the *R* to a CSV file and output it.

Although the time complexity of Algorithm I, without using parallel processing, is *O(n)*, *n* is an extremely large number in our case (e.g., 30 million review texts and 0.7 million products in the categories of *Electronics*). Through using package "*multiprocessing*" in Python, the asynchronous parallel processing is implemented in Steps 2 and 4. Different from synchronous execution blocking the main program to ensure the processes are completed in the same order in which it was started, asynchronous does not consider the sequence of the results of the processes and usually finishes the task faster. The output review data file is used in the latter stage of training the sentiment classification model. The time complexity of Algorithm I using parallel processing is *O(n/c)*, where *c* is the number of CPU cores used in computing.

B. Production feature extraction from online reviews

To make full use of the features and sentiment tendencies of the input reviews in our approach, we propose an NLP-based method to extract product features and retrieve corresponding texts that only represent the selected features and the irrelevant information is eliminated. The proposed method is illustrated in Fig. II. The detail of each step is explained in the following.

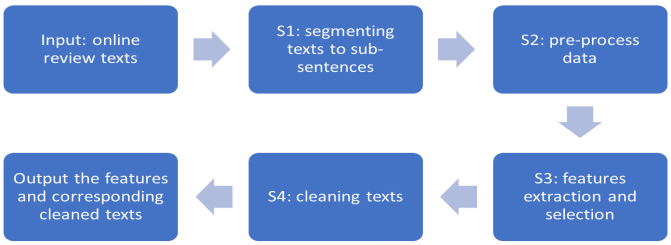

Fig. II. The process of production feature extraction

Step 1: Separate each review text into sub-sentences. A customer's review may include multiple sentiments (e.g., positive, neutral, negative) to a product's different features. Therefore, we cut each of the input reviews into sub-sentences to minimize the number of features mentioned in a single text fragment (i.e., a sub-sentence) as much as possible. Ideally, each sub-sentence only describes a particular product feature. In more detail, the reviews are normally short texts with one or two paragraphs due to the concise and brief style of online reviews. The words expressing strong feelings or sentimental tendencies of a particular feature in a review are usually followed by words or sentences which might be redundant text and not be associated with customer tendencies, such as commonly detailed feature descriptions. This redundant information can introduce noise to our approach when using

[1] It can be noticed that the size of the JSON files is quite large for a single machine, especially a personal computer. Reading the data from them requires considerable memory that a personal computer may not fulfil. Furthermore, filtering data based on the requirements to create the training dataset on a single machine with a single running thread is time-consuming due to the enormous number of review texts and products. To solve these two problems, we use the High-Performance Computing (HPC) cluster to read the data and implement parallel processing during the data filtering.

[2] JOSN was driven from JavaScript but many modern programming languages include code to generate and parse JSON-format data, which means the JSON is a language-independent data format.

text classification models to identify sentiment tendencies.

Step 2: Pre-process the data: tokenize a sentence to words and then use NLTK[3] for part-of-speech tagging.

Although Amazon does not provide an emojis input option on its review system, some reviewers still could input emojis to present their sentiments via the tool provide by the input methods installed in phone or PC systems. Since emojis convey a fruitful sentiment tendency and information, understanding the meaning of the emojis during sentiment analysis is very useful [62]. In our approach, we use a python package called "emot[4]" to identify the emojis from the input product reviews and convert them to the description texts, for example, ':-)' to "*Happy face smiley*".

To efficiently identify the most representative words as features from the review texts, we only keep noun words and delete the rest under the 1-gram method or keep noun and adjective words under the 2-gram method (the details are explained in *Step 3* and *Note 1*). The reason for this is that almost all features extracted for product ranking are nouns from the aspect of semantics [2], [3], [5]–[8], therefore we do not need to consider the words which may contain sentimental tendency in the current step. For the convenience of word count in the latter stage, we convert all the noun words to lower case. To minimize the noise and improve the efficiency, the Porter stemming algorithm [63] is used to reduce words to their root form, e.g., *swimming* will be stemmed to its root form, *swim*, after applying this algorithm. Some sentences may be totally deleted since they may not contain any noun word. An index is then allocated to each sentence, indicating which review the sentence belongs to. There could be more than 1 sentence having the same index if all sub-sentences are taken from the same review. Table I shows examples of texts after these pre-processing steps. Due to the space limitation, we provided the corresponding original review texts of Table I in Section D of the Appendix to help the reader better understand our method.

TABLE I
EXAMPLE RESULTS OF REVIEW TEXT PRE-PROCESSING

Index	Result texts
1	samsung box people
1	store screw wall mount inform booklet
1	setup painless friend tv wall footage enthusiast care quality look flaw picture
…	…
1	pack job hole box back mark present
2	input lag samsung game

Step 3: Feature extraction and selection: apply the bag-of-words and Term Frequency-Inverse Document Frequency (TF-IDF) algorithms to transform words into sparse feature vectors. We can use the *CountVectorizer* class implemented in the scikit-learn package[5] to construct the vocabulary of the text documents obtained in Step 2, then count the word frequency in respective text documents, and transfer it to term frequency-inverse document frequency (*tf-idf*). To filter out the feature words with lower frequency from the review documents, the invited experts who will carry out the product ranking in the later stage will give a threshold which indicates a minimum word frequency or a sum of *tf-idf*. A sum of word *tf-idf* in the feature vectors represents the value of the word total frequency divides its *idf*: $\sum_{d=1}^{n_d} tf\text{-}idf(t,d) = \sum_{d=1}^{n_d} tf(t,d) \times \log\frac{n_d}{1+df(d,t)}$ (The concept of *tf-idf* is reviewed and demonstrated in the Appendix file Section B). Noun words whose sum of frequency or *tf-idf* in the feature vectors are above the threshold are kept in our dictionary. The experts choose the most important feature words and eliminate irrelative words concerning the products. In more detail, the invited experts select the most representative words from the list of high frequency or summed *tf-idf* words for a feature and regard the selected words as a set of linguistic terms for the feature.

Note 1. In NLP, the contiguous sequence of items like words, letters, or symbols is called n-grams [60]. When applying the *CountVectorizer* to count the word frequency, the 1-gram and 2-gram are both implemented in our approach. The 1-gram means each item or token in the vocabulary represents a single word. The 2-gram is also applied to avoid missing product attributes. Observing that some noun phrases include adjective (JJ) words, e.g., '*operational performance*', we keep the JJ word to ensure accurate noun phrase tagging.

Note 2. Due to the personalized expression from customers, different keywords could be used to express the same attribute in a text review. For example, the attribute "price" could have the following relative keyword: "money", "value for money", "expensive", "costly". Most of these related words can be obtained by our model, e.g., money can be obtained by the *1-gram with noun* method, "value (for) money" can be obtained by the *2-gram with noun* method ("for" is an English stop word and it will be deleted. Thus, the word is detected as "value money"), and JJ method. The *1-gram with JJ* method can obtain JJ properties such as "expensive" and "costly".

Note 3. Online product ranking is one kind of decision-making problem. Nowadays, the decision environment such as online shopping platforms is becoming highly complex with the increasing social and economic development [2]. It is difficult for a single decision-maker to obtain an optimal solution to a complex decision-making problem. Hence, serval experts with related professional knowledge and experience are always invited to reach an optimal solution through dynamic discussion and learning [64]. In our case, the experts are invited to identify the product features from our extracted high-frequency words and create a feature corpus for this product. Once the corpus is created, it can be used to automatically select useful feature words and filter out irrelative words for similar products (in the same product sub-category) without the extra need for experts' involvement. Also, note that the usefulness of this corpus is not limited to our work in this paper. Its usefulness can be extended to more scenarios. For example, the feature words in the corpus could be the keyword for customer search and product recommendations in a shopping platform. In practical

[3] NLTK: https://www.nltk.org/
[4] https://github.com/NeelShah18/emot
[5] Scikit-learn: https://scikit-learn.org/stable/

applications such as integrating our approach to an online shopping platform, these experts from linguistics could create a feature corpus for a new product without the reviews (e.g., no sales happen yet). Experts from e-commerce could then use our feature extract method to create or update the corpus for the product with a certain number of reviews.

Step 4. Cleaning text by only keeping retrieved feature texts from reviews. After the experts' selection, several sets of linguistic terms of feature (each feature represented by one or more words/phrases) could be obtained. Then, we clean the review texts by only keeping the sentences which represent sentiment tendencies for each feature from the original review text. This can be achieved by the obtained tokenized sentences listed in Step 2 (e.g., Table I) and the feature vector in Step 3 (e.g., Table A.I in Appendix B). Based on the latter one, the reviews that contain the linguistic terms of the selected feature can be located by their text indexes. Then for each of these reviews, we can get index(es) of the sentence(es) that contain the selected feature by segmented sentence list (e.g., Table I). For the resulting sentences having the same review index, we could combine these sentences together since they are from one review. The new list of review texts is then sent to our trained text classification model and returns the probability distribution on class label set such as *{"negative", "neutral", "positive"}*.

Note 3. Since we cut review texts to sentences at the start, the risk of multiple features being mentioned in a single text from our output list is decreased. Therefore, the noise texts for the selected feature are eliminated as much as possible before the sentiment analysis process. The corresponding cleaned texts for each feature are directly derived from the texts of the original reviews not from texts obtained from Step 2. Therefore, all types of words are included in the cleaned texts, such as adjectives and adverbs. These kinds of words cover a huge part of the sentimental values of the feature word. In addition, the particular word indicating the relationship between the other two words, which has an enormous impact if our method can successfully capture the correct sentiment expressed in a sentence is also included.

Example 1. The following provides an example regarding the output of reviews from the product "Samsung Flat 55-inch 4k 8 Series UHD smart TV with HDR and Alexa Compatibility" in Amazon.com[6]. Since LDA is a commonly used statistic feature-extracting method used in current research of product ranking based on online reviews, we compare our proposed feature-extraction method with it. The results are shown in Table II. The output of LDA is under the setting of having 5 topics, removing English stop words, and excluding words that occur too frequently across reviews (top 10 percentage of frequency words which normally contain no or very less useful or discriminatory information), these parameters are optimized by trial-and-error strategy. From Table II, we can see the words for some of the common features of TV to affect the user's ratings are identified by our method: "picture" and "colour" for picture quality. In contrast, the feature is hard to guess based on reading the five most important words of each topic output by LDA. There are much fewer product feature-related words identified by LDA than by our method. That is because the LDA is a topic modelling method that has better performance with long texts that have different topics. However, online reviews of products are short and initially limit the topic to focusing on one item. Furthermore, LDA can only extract features from reviews but doesn't provide the functionality to keep the information of which sentences contain these features, which is inconvenient for analysing product features with PLTSs. All these imply that the LDA is not suitable for features and corresponding texts extraction.

TABLE II
COMPARISON OF FEATURE EXTRACTION RESULTS BETWEEN OURS AND LDA

Our method			**LDA**	
Words	*tf*	*tf-idf*	Index	Top 5 important words
picture	123	34.43	Topic 1	box home room new movie
tv	113	33.60		
samsung	105	27.08	Topic 2	settings does refresh rate turn
great	81	35.11		
set	67	17.99	Topic 3	oled color beautiful did best
remote	65	18.87		
quality	55	18.39	Topic 4	used work little clear recommend
app	48	13.92		
smart	45	13.75	Topic 5	settings audio hdmi menu work
good	42	16.80		
sound	41	12.00		
feature	38	12.09		
color	36	12.77		
screen	32	11.53		

C. Backbone text classification models

In the previous two subsections, we obtained the training data for a specific category of product and extracted feature texts for different features from online reviews of two or more products under this category. This subsection presents how we need to implement the grey parts in Fig. I – training a DL-based text classification model and inputting these feature texts to generate PLTSs on different features for these products. Therefore, the obtained PLTSs can be used to rank products based on different features. In this subsection, the following problems are addressed: 1) how to match review rating labels with linguistic terms? 2) is it reasonable to use such a trained DL-based model to generate PLTSs by inputting extracted feature texts?

Since most of the online-shopping websites ask customers to rate products in the range of 1 to 5 stars or scores, the sentiment words of text reviews can be dived into 3 parts {*"negative", "neutral", "positive"*}, or 5 parts *{"very negative", "negative", "neutral", "positive", "very positive"}*. This is deemed as a linguistic term set (LTS) and each part of it is a

[6] https://www.amazon.com/Samsung-UN55RU8000FXZA-FLAT-UHD-Smart/dp/B07NC9XWG5/ref=sr_1_2?keywords=Samsung&qid=1575553046&s=tv&sr=1-2&th=1

linguistic term (LT). As aforementioned in the Introduction, the probabilistic distribution on all labels or classes from the classification result can match the mathematical form of PLTSs. That is because the LTs such as "*neutral*" can be directly formulated as labels for the classification models. Therefore, the output of a text classification model can be considered as PLTSs straightforwardly in our approach. For example, the output PLTS for the input sentence of "disappointed with the picture quality" is *{"negative" (0.33), "neutral" (0.66), "positive" (0.01)}*.[7] In this subsection, the implementation of state-of-the-art DL-based models to generate PLTSs is detailed. In our approach, we use the review dataset obtained in Section III.A to train DL-based text classification models. Then, we apply these trained models to carry out the sentiment analysis for the input texts, that is inputting the review texts of each selected product feature (obtained from the original input reviews by Section III.B) to predict the probability distributions among all labels. Fig. III depicts the unified problem of multi-class text classification in our approach.

Note that how to match review rating labels (star/score) with linguistic terms is still an open problem. As said before, the exiting work [19], [21], [22] divided reviews as "*positive*" or "*negative*" to match with linguistic terms, which was not accurate enough and only provided a rough sentiment tendency. Another straightforward solution is to match them with 5 parts' LTS since most websites such as Amazon.com ask customers to rate products in a range of 1-5 stars/scores. We have already analyzed in Section I.A that this method is not reasonable. The sentiment intensities represented by the rating scores are not equivalent to the sentiment intensities hidden from the unstructured review texts. A rating score is only a single number to summarize the overall satisfaction of a product/service, but the sentimental words used to describe the satisfaction level of different aspects of a product are far more informative and fine-grained. These sentimental words can be distributed in review texts with different rating scores, like the example shown in the Introduction. Based on these reasons, we argue that rating score labels can only represent the sentiment tendencies and rough sentiment intensities but cannot precisely reflect the sentiment intensities. Therefore, matching rating labels with linguistic terms directly cannot provide accurate PLTSs in identifying the correct sentiment tendencies from online reviews. In our approach, we use the overall review texts to train a text classification model, and then to generate PLTSs for the input feature texts (extracted from previous steps), because most of the online website does not have a customized rating system for different categories of products to rate on different features. To ensure prediction accuracy and training efficiency, it is essential to relabel the training review texts to optimize the supervised information in the training dataset. In this scenario, classifying the level of sentiment tendency for all rating score labels based on their corresponding review texts and matching them with appropriate linguistic terms needs to be implemented before training the model in our approach.

Another research question which might also be interesting is, how the performance of DL-based text classification models for the texts of extracted features. Our idea of using the same trained model to classify sentiment tendencies for the extracted feature texts is similar to transfer learning (TL). TL is a ML problem that focuses on storing gained knowledge while solving one problem and applying it to a different but related problem in the same or similar domain [65]. In our approach, we borrow the similar idea from TL. We believe more and more robust prior knowledge is gained by the model while using the big data obtained in Section III.A to do the training [66], [67]. After the training process, the model is equipped with strong prior knowledge to do the sentiment classification for the texts of extracted features without the need for further feature-text-specified training. This is because the domain of the new problem (sentiment classification for the extracted feature texts) is the same as the original one, both focusing on analyzing review texts from the similar products. More precisely, we use the trained model to solve a new problem that is the sub-problem of the original one, which is a different part from TL.

After obtaining PLTSs from the review texts of products, we could rank these products by related extracted product features. This is a typical multi-criteria decision making problem [2]. In Section V, we show how to apply the PLSTs generated by our approach to the problem of product ranking based on online reviews.

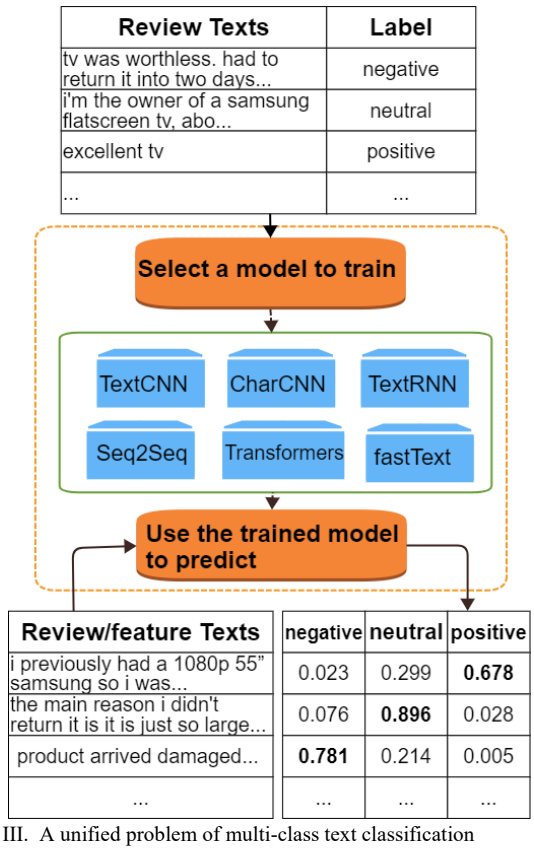

Fig. III. A unified problem of multi-class text classification

[7] It is noted that the PLTSs extracted from text reviews from our approach are complete, that means, $p_s = 0$. Hence, we simplify the PLTS as $h_S(p) = \{s_\alpha(p_\alpha)|\alpha = \{0,1,\dots,q\}\}$ in this study.

IV. Experiments for Sentiment Classification To Produce PLTSs

Although the performance of all mentioned DL models for text classification has been tested in many studies, in order to find the most suitable backbone models for our approach, it is necessary to design and implement experiments to test the performance of various DL models in our scenario. We use the metrics, including *accuracy*, *precision*, *recall*, and *F1-score* used in previous studies [35], [60] to evaluate the performance of each model mentioned in Section II, including TextCNN, CharCNN, TextRNN, Seq2Seq, transformer, and fastText. The last three metrics are shown in Appendix file Section C.

A. Experimental Settings

To prepare the training data, we extract the review data of all TVs under the category of TV from the Amazon review dataset [61]. For all labels (rate scores in the range of 1 to 5), the numbers of review texts are around 25k, 9k, 13k, 36k and 116k, respectively. Based on this dataset, all the models are trained and well fine-tuned before applying the models to the test set and analyzing results by grid search. We also extract the three features *picture quality*, *remote control*, and *sound quality*, and the corresponding feature texts (292 in total) from the reviews (353 in total) of TVs "Samsung Flat 55-inch 4k 8 Series UHD smart TV with HDR and Alexa Compatibility" and "LG 65SM8600PUA Alexa Built-in Nano 8 Series 65 4k Ultra HD smart LED Nano Cell TV"[8]. Two TVs will be ranked in Section V based on PLTSs of three extracted features. Therefore, we use these 292 feature texts to perform the final test for the trained different text classifiers.

TABLE III
General Parameter Setting for Different Classifiers

Classifier	*Optimizer*	Activation & Loss function	*Initial LR*	Drop rate	Maximum epochs	Dynamic LR
TextCNN	SGD	ReLU & NLL	0.3	0.8	20	Y
CharCNN	Adam	ReLU & cross-entropy	0.001	0.5	100	Y
TextRNN	SGD	N/A & NLL	0.75	0.8	60	Y
Seq2Seq	SGD	N/A & NLL	0.3	0.8	20	Y
Transformer	AdamW	N/A	4e-5	N/A	1	Y
fastText	N/A	N/A & SoftMax	1	N/A	25	Y

Table III describes some of the general parameter setting for different classifiers. The rest of the parameters of each classifier are depicted as follows. For the model of TextCNN, pre-trained Glove Embeddings [68] for encoding words are used, and the vector size of Glove Word Embeddings is 300. The learning rate (LR) is set to 0.3 and reduced by a factor 0.5 after every 1/3 of the maximum epochs. The model can always converge after around 10 epochs for all test experiments; thus, we finally set the maximum epochs to 20. The dropout rate with kept probability is set as 0.8 to prevent the over-fitting during the training.

For the model of CharCNN, the initial learning rate is set as 0.001 and it is halved at every 3 epochs. The maximum epochs and dropout rate are set as 100 and 0.5, respectively.

In the TextRNN, 2 layers of bi-LSTM are set and each one has 32 hidden units. The maximum sequence length for bi-LSTM is set as 30. The initial learning rate is 0.75 and it is halved after every 1/3 of the maximum epochs. The maximum epochs and dropout rate are set as 60 and 0.8, respectively. Pre-trained Glove Embeddings for encoding words are used in the model, the vector size is 300.

We use single bi-LSTM with 32 hidden units as Encoder in Seq2Seq, and set no restrictions with the sequence length, which means that the sequence length is determined by batch. Pre-trained Glove Embeddings for encoding words are also used. The settings of the learning rate, maximum epochs, dropout rate and vector size for word embedding are the same as those in TextCNN.

In the Transformer, a pre-trained model on English language BERT-based-cased is employed. The initial learning rate is set as 4e-5 and the number of training epoch is 1. L2 penalty for the weight decay is added to the model to prevent overfitting. The "AdmaW" optimizer is employed to decouple the weight decay from the optimization step, which means, the learning rate and weight decay are optimized, separately. The maximum sequence length supported by the model is 128.

In the fastText, each word is treated as composed of n-grams when preparing the word vectors. We set the word n-gram as 2, the vector size for word Embeddings is 100. The initial learning rate and the maximum number of epochs are set as 1 and 25. The SoftMax function is employed as the loss function.

All the results shown in the following subsections are the mean of experiment results from the multiple running (3-5 times). The reason for it is to avoid possible bias. The results from the different runs are quite close and the result variance is small. The potential reason is that the dataset used in our experiments is large enough to eliminate the bias and another possible reason is all the DL-based models we implemented are well-designed, robust and confident enough to make similar decisions in every run. Also due to the space limitation, we only give the mean value here. Our approach is implemented in PyTorch [69] and the source code and our experiment data are publicly available[9].

B. Experiment of Matching Rating Labels with Linguistic Terms

In this subsection, we first test the way of matching the rating scores of 1-5 with the 5 parts' LTS. Each rating score is a label in the text classification model. Two datasets derived from the Amazon TV dataset are used to train all models: 20k and 40k reviews in total, in which each label contains 4k and 8k reviews, respectively. The test dataset used in the experiments contains 1.5k review texts for each label and 7.5 k in total. The reason why we use two different size datasets is, the review datasets for different products derived from the Amazon review dataset have different sizes and therefore we want to investigate the performance of different models under different numbers of training review texts. We do not use the whole TV review

[8] https://www.amazon.com/LG-65SM8600PUA-Alexa-Built-NanoCell/dp/B07PQ97CRW/ref=sr_1_1?keywords=LG&qid=1575553310&s=tv&sr=1-1

[9] https://github.com/liulei1260/A-DL-based-Sentiment-Analysis-Approach-for-online-producting-ranking-with-PLTSs

dataset to train the model because it is unbalanced. Using balanced training and test dataset can prevent a false sense of a highly accurate model [70]. The disparity of classes in the variables will mislead the algorithm to category the test instance into the class with more instances (majority class) in a high probability. This kind of model cannot predict the rare event, the minority class, but still get a higher predict accuracy [70], [71].

The experiment results are listed in Table IV. From this table, we can see the Transformer achieved the best performance in both experiments. However, it should be pointed out that the best performed model only gets 62.57% and 57.18% of training and test accuracy, which implies that even the best model cannot perform a precise classification. By doubling the number of training data does not increase the training and test accuracy much for each model, which means the lower accuracy is not relevant to the scale of the training dataset.

TABLE IV
TRAINING AND TEST ACCURACY OF EACH MODEL IN DIFFERENT DATASET FOR 5 PARTS LTS

Classifier	*Train AC* (%) ↑		*Test AC* (%) ↑	
	20k	*40k*	*20k*	*40k*
TextCNN	45.44	43.05	39.38	39.76
CharCNN	<u>60.44</u>	<u>61.73</u>	43.46	45.00
TextRNN	46.17	43.24	41.36	40.36
Seq2Seq	42.56	45.23	31.06	36.02
Transformer	**61.17**	**62.57**	**56.32**	**57.18**
fastText	53.26	53.61	<u>51.38</u>	<u>52.00</u>

Note. AC in Table IV is short for accuracy. 20k or 40k in the fourth or fifth column means the test *AC* of 7.5k test dataset under different models trained by 20k or 40k dataset. The best training/test result for each dataset (each column in the table) is in **bold,** and the second-best result is underscored.

Table V shows the detailed results of each label for different models under the 40k training dataset. From Table V, we can see the performance of identifying the review texts with scores 2, 3, and 4 is much worse compared with scores 1 and 5 for all models. It could be explained by the fact that the boundaries of the data in these three classes are not clear. Besides, even for scores 1 and 5, the accuracy is not high enough, only at around 60%. Naturally, the text reviews with extreme bad or good feedback should be classified with high accuracy. The lower accuracy of score 1 (or 5) might be due to the review texts with extra bad (or good) feedback may be distributed in both scores of 1 and 2 (or 4 and 5), in the sense that a relatively unclear boundary between classes causes the difficulty of multi-class classification with more classes. Some reviewers are critical (merciful) and hard to give a score of 5 (1) even with really positive (negative) feedback on a product.

Based on the above analysis, we could see that matching the rating score of 1-5 with 5 parts' LTS cannot train a text classification model to identify correct sentiment tendencies from online reviews with a high accuracy, which validates our previous assumption. Therefore, matching the rating scores from 1-5 with correct linguistic terms and then classifying different sentiment tendency levels for the review texts, is the problem we need to solve.

TABLE V
RESULTS OF EACH LABEL FOR DIFFERENT MODELS IN 40K TRAINING DATASET

Classifier	*Test Accuracy* (%) ↑	*Metric*	*Score 1*	*Score 2*	*Score 3*	*Score 4*	*Score 5*
TextCNN	39.76	*Precision* (%) ↑	**44.63**	0	31.92	0	<u>*43.72*</u>
		Recall (%) ↑	<u>*62.20*</u>	0	57.30	0	**75.30**
		F1 (%) ↑	<u>*53.32*</u>	0	41.00	0	**55.32**
CharCNN	45.00	*Precision* (%) ↑	<u>*56.41*</u>	38.36	35.72	35.40	**57.29**
		Recall (%) ↑	**59.80**	34.50	35.90	35.90	<u>*58.90*</u>
		F1 (%) ↑	**58.58**	36.45	35.81	35.65	<u>*58.08*</u>
TextRNN	40.36	*Precision* (%) ↑	38.98	30.49	34.76	<u>*39.34*</u>	**51.30**
		Recall (%) ↑	**74.70**	11.80	31.50	26.60	<u>*57.20*</u>
		F1 (%) ↑	<u>*51.23*</u>	17.01	33.05	31.74	**54.08**
Seq2Seq	36.02	*Precision* (%) ↑	**53.74**	25.00	26.09	0	<u>*41.28*</u>
		Recall (%) ↑	**63.90**	0.10	70.6	0	<u>*45.50*</u>
		F1 (%) ↑	**58.38**	0.19	38.11	0	<u>*43.29*</u>
Transformer	57.18	*Precision* (%) ↑	<u>*64.62*</u>	45.57	48.41	59.28	**66.86**
		Recall (%) ↑	<u>*66.50*</u>	46.90	47.50	47.90	**77.10**
		F1 (%) ↑	<u>*65.54*</u>	46.29	47.95	52.98	**71.62**
fastText	52.00	*Precision* (%) ↑	<u>*60.28*</u>	43.53	43.53	48.39	**64.63**
		Recall (%) ↑	<u>*59.50*</u>	43.10	46.10	46.60	**64.70**
		F1 (%) ↑	<u>*59.88*</u>	43.31	44.77	47.47	**64.66**

Note. The label with the best result in each model (each row in Table IV) is in **bold,** and the second-best is in underscored *italic*.

Our solution for this problem is to match the rating scores 1-5 with 3 parts' LST *{"negative", "neutral", "positive"}*. We investigated three possible combinations of matching solutions that could help to train a much better prediction model in our experiment: 1) *{"negative": (1, 2), "neutral" (3), "positive": (4, 5)}*; 2) *{"negative": (1, 2), "neutral" (3, 4), "positive": (5)}*; 3) *{"negative": (1), "neutral" (2, 3), "positive": (4, 5)}*. We do not test the combination of *{"negative": (1), "neutral" (2, 3, 4), "positive": (5)}* since the texts in data sets of labels 2 and 4 have opposed sentiment tendencies naturally. Three new datasets are created by re-labelling all 180k TV review texts based on 3 different matching solutions. The training dataset for each matching solution contains 24k review texts in which each label from *negative, neutral,* and *positive* contain 8k review texts. Each corresponding test dataset contains 1k review texts for each label and 3k in total. Table VI lists the training and test accuracy of each model under different matching solutions.

Compare with the result in Table IV, we could see that the training and test accuracy of all models in Table VI are increased by around 20%. Most of the models have the best results under the matching solution 3. All models get a better test accuracy of higher than 60% under this solution, especially

for the Transformer and fastText which achieves 74.90% and 73.37%, respectively.

TABLE VI
TRAINING AND TEST ACCURACY OF EACH MODEL FOR 3 MATCHING PLANS

Classifier	*Train AC* (%) ↑			*Test AC* (%) ↑		
	S 1	*S 2*	*S 3*	*S 1*	*S 2*	*S 3*
TextCNN	67.06	66.15	**68.01**	60.23	61.27	*61.57*
CharCNN	81.66	82.75	**83.01**	65.90	65.53	*68.33*
TextRNN	68.90	67.30	**69.68**	63.13	61.87	*64.60*
Seq2Seq	69.09	73.18	**73.32**	57.47	57.13	*62.80*
Transformer	**83.26**	82.05	83.20	75.97	*76.03*	74.90
fastText	78.09	**78.32**	78.29	71.43	71.97	*73.37*

Note. The best training result for each model (each row in the table) is in **bold,** and the second-best test result is in underscored *italic*.

Table VII shows the detailed results of each label from the LTS {"negative", "neutral", "positive"} for different models under matching solution 3 where the label with the best result in each model (each row in the table) is in **bold**. The table shows that the class of *positive* and *negative* can be categorised with around 70% accuracy in all models and some of the models could even reach 80% accuracy. Although the prediction accuracy for the label of *neutral* is around 60%, considering the data distribution of three labels in the TV dataset is 11.91%, 11.89%, and 76.23%, we could be assured that the minority class cannot lower the overall prediction accuracy too much. Therefore, all models under matching solution 3 could get a better prediction accuracy for the real-world unbalanced data.

TABLE VII
RESULTS OF EACH LABEL FOR DIFFERENT MODELS IN 24K TRAINING DATASET

Classifier	*Test AC* (%) ↑	*Metrics* (%)	*negative*	*neutral*	*positive*
TextCNN	61.57	*Precision* ↑	61.46	52.08	**68.36**
		Recall ↑	**66.50**	53.70	64.50
		F1 ↑	63.88	52.88	**68.36**
CharCNN	68.33	*Precision* ↑	71.11	54.60	**78.73**
		Recall ↑	70.09	53.40	**80.70**
		F1 ↑	71.00	53.99	**79.70**
TextRNN	64.60	*Precision* ↑	65.71	54.93	**74.14**
		Recall ↑	67.10	57.30	**69.40**
		F1 ↑	66.40	56.09	**71.69**
Seq2Seq	62.80	*Precision* ↑	68.62	49.92	**76.77**
		Recall ↑	**66.70**	63.20	58.50
		F1 ↑	**67.64**	55.78	66.40
Transformer	74.90	*Precision* ↑	73.90	65.39	**86.07**
		Recall ↑	78.80	61.80	**84.10**
		F1 ↑	75.84	63.64	**85.07**
fastText	73.32	*Precision* ↑	75.25	62.52	**82.44**
		Recall ↑	73.60	63.40	**83.10**
		F1 ↑	74.41	62.95	**82.76**

In summary, the method of matching the rating scores of 1-5 with 5 parts' LTS in the current work cannot guarantee a text classification models to identify the correct sentiment tendencies from the online reviews with a high accuracy. All the experiment results in Tables IV-VII provide sufficient support that our matching solution can solve this problem and benefit the training of a text classification model to identify sentiment tendencies from review texts.

C. Experiment on the models and the number of training data

In this part, we compare the performance of different models trained by different sizes of training text datasets under matching solution 3. Different types of products may have different numbers of text reviews. The effect of different sizes of training data on each model needs to be investigated. Four datasets are extracted from the TV review dataset and used to train all models: 12k, 24k, 36k, and 48k reviews in total in which each label contains 4k, 8k, 12k, and 16k review texts, respectively. The test dataset used in this experiment is unbalanced that contains 353 review texts from Samsung and LG TVs mentioned before. As a comparison, a balanced test dataset derived from our TV dataset is also used that contains 1k review texts for each label and 3k in total. The experiment results are listed in Table VIII where the best result in each column is highlighted in **bold** and the second-best in underscored.

From Table VIII, we can see Transformer achieves very competitive performance for both balanced and unbalanced datasets with 75.13% and 82.72% accuracy, respectively. fastText has the second-best results on the balanced dataset while CharCNN achieves the second-best results on the most of experiments for the unbalanced dataset. Having a closer look, we can find that CharCNN has a higher accuracy for classifying data with labels of *negative* and *positive* which are majority classes in real data. The rest of the models also have acceptable results on both two datasets, at around 65% accuracy. As a comparison, we also investigated the use of Stanford CoreNLP toolkit on our data. However, it only gets 10.19% accuracy on the original "Samsung and LG TV" dataset. Although after applying our matching solution method, the accuracy only increased to 55.52%, which is still much lower than our experimented DL-based models. This indicates the current works which use the rule-based sentiment analysis methods cannot generate accurate PLTSs from the review texts.

By comparing the performance of each model under different training datasets, we find the test accuracies of TextCNN, CharCNN, and Transformer increase as the size of the training data increase. By contrast, the test accuracies of TextRNN, and Seq2Seq even decrease when the training data is increased in some cases. Different from others, the test accuracy of fastFast is more robust with respect to the number of training data.

In summary, we would suggest using Transformer as the first choice to train a sentiment analysis model to generate PLTSs from online reviews. Its prediction accuracy can be slightly increased with the increase of training data. The classifier of fastFast would be the second choice since it can output a competitive accuracy result in a short time and less training data.

TABLE VIII
EXAMPLE FEATURE VECTORS FOR A TEXTS SET

Classifier	*Test AC on test dataset (%)* ↑				*Test AC on unbalanced dataset (%)* ↑			
	12k	*24k*	*36k*	*48k*	*12k*	*24k*	*36k*	*48k*
TextCNN	59.67	61.57	62.10	61.80	59.77	62.89	64.87	67.42
CharCNN	64.27	68.33	69.10	69.20	67.71	*76.20*	*76.77*	*76.79*

TextRNN	61.77	64.60	64.37	64.67	68.27	64.59	71.95	66.01
Seq2Seq	57.10	62.80	64.80	65.73	66.01	73.35	72.52	67.14
Transformer	**73.80**	**74.90**	**75.07**	**75.13**	**79.60**	**80.45**	**82.15**	**82.72**
fastText	72.20	73.37	72.87	74.80	75.64	74.79	73.09	71.67

D. Experiment of sentiment classification for extracted feature texts

From the literature reviews in Section I.A, we know that DL has been proven to improve the defects of traditional ML methods on text classification. However, the research on implementing DL models for generating PLTSs has not been well-studied, due to lacking large training data and methods of matching rating labels of review texts with correct linguistic terms. After solving these two problems, it is also essential to conduct a comparative analysis of the performance between DL and ML models for generating PLTSs under our approach. In addition, it is necessary to experimentally prove our proposed assumption in Section III.C, that is, the trained DL model in our approach has strong prior knowledge to perform the sentiment classification for extracted feature texts without needing any further training. Therefore, in this section, we test the performance of different DL and ML classifiers in our approach by using the extracted feature texts from product reviews. Same as before, we use the Grid Search to optimize the parameters for all ML classifiers. There is no ground truth label for this dataset since all feature texts are only part of their original review texts. Therefore, the sentiment tendency of each feature text piece may be different from the label of its original text. To solve this problem, we invite three linguistic experts to manually annotate the texts (292 in total) of three extracted features, *picture quality*, *remote control*, and *sound quality*, from the "Samsung and LG TVs" dataset. They vote for each feature text and the label with the majority vote will be the final annotation result for this text. Experts will make a discussion and vote again when facing the same vote for all labels of a feature text, which is rare in our case. Since all three experts came from Sichuan University with an English language and literature study background, their annotations of these feature texts can be considered as ground truth labels. Specifically speaking, the identified level of sentiment tendency from the LTS {*"negative", "neutral", "positive"*} for each feature text is considered as the ground truth label. All classifiers are trained by the balanced dataset of 24k reviews from Section IV.C, then tested on the 3K test dataset and 292 feature texts (third and fourth column of Table IX respectively).

From Table IX, we can see Transformer achieves the best performance (90.03% accuracy) and fastText has the second-best (76.98%) on the feature texts. Except for CharCNN and Seq2Seq, most of the classifiers achieve better performance on the feature texts (4th column) compared with their performance on the 3k test dataset (3rd column). The feature text normally contains only one kind of sentiment tendency since it is a small part of original review texts and is usually very short. Therefore, with enough knowledge obtained from the 24k review dataset, all the models could obtain a good test accuracy on feature texts. Both the fastText and Transformers obtain high classification accuracies, and their training time was short (0.7 and 3.55 minutes respectively). In contrast, the training time of TextCNN, TextRNN, Naive Bayes and KNN was about 1-5 minutes which is relatively shorter, however, their prediction accuracies are much lower than the rest. Compared with most of DL classifiers, all conventional ML classifiers obtained much lower accuracies on both test datasets, (the difference is around 5-15%). All of these can be because the state-of-art DL models can improve the defects such as data sparsity, dimension explosion, and poor generalization ability of traditional ML methods and better handle the large dataset on text classification.

TABLE IX
RESULT OF EACH MODEL IN REVIEW TEXTS OF EXTRACTED FEATURES

Classifier	*Method*	*Test AC* (%) ↑ on 3K test dataset	*Test AC* (%) ↑ on 292 feature texts	*Training time (mins)* ↓
TextCNN	DL	62.89	71.82	0.9
CharCNN	DL	76.20	73.20	12.68
TextRNN	DL	64.59	73.54	3.25
Seq2Seq	DL	73.35	69.07	18.28
Transformer	DL	**80.45**	**90.03**	3.55
fastText	DL	74.79	76.98	**0.7**
XGBoost [72]	ML	70.67	67.70	10.85
Naïve Bayes	ML	69.83	62.20	1.71
SVM	ML	69.16	64.26	30.25
KNN	ML	58.37	61.51	5.7

In summary, Table IX proves our claim that the trained DL model in our approach has strong prior knowledge to perform the sentiment classification for extracted features texts without needing any further training.

V. CASE STUDY

In this section, we apply the experimental results of the unbalanced test dataset containing 353 review texts to rank the Samsung (type name: UN55RU8000FXZA) and LG TV (type name: 65SM8600PUA). According to our approach, the original input review texts, and the extracted review texts of the three features, "*picture quality"*, "*remote control"*, "*sound quality*", are sent to the trained DL-based model for text classification. The results are output in the form of PLTSs which contain the overall sentiment tendencies and corresponding probability distribution of each review text and that of each feature contained in each review text. Compared with the sentimental distribution of the overall review text, the sentiment distribution of the feature contained by the text is more commonly used in ranking products because different features usually have different importance and weight.

In this study, we represent the sentiment distribution of the three features in the test dataset in the form of PLTS and then rank the products. Let Samsung and LG TV be A_1 and A_2, respectively, which are measured by the three features, "*picture*

quality" (c_1), "*remote control"* (c_2), "*sound quality*" (c_3). The PLTS $h_S^{ijr}(p) = \{s_\alpha(p_{\alpha,ijr}) | \alpha = \{0,1,2\}\}$ represents the performance of product A_i ($i \in \{1,2\}$) with respect to feature c_j ($j \in \{1,2,3\}$) commented in the review text t_r ($r \in \{1,2,\dots,n\}$). First, we integrate the PLTSs $h_S^{ijr}(p) = \{s_\alpha(p_{\alpha,ijr}) | \alpha = \{0,1,2\}\}, r = 1,2,\dots,n$, into a collective one $h_S^{ij}(p) = \{s_\alpha(p_{\alpha,ij}) | \alpha = \{0,1,2\}\}$ to reflect the group opinion on product A_1 with respect to feature c_j. Each review text is regarded as equally important. Considering that each text does not always mention all the three features, Eq. (2) is used to integrate the probability $p_{\alpha,ij}$ of s_α in $h_S^{ij}(p) = \{s_\alpha(p_{\alpha,ij}) | \alpha = \{0,1,2\}\}$ [2].

$$p_{\alpha,ij} = \frac{1}{Q_{ij}} \sum_{r=1}^{n} p_{\alpha,ijr} \quad (2)$$

where Q_{ij} is the number of reviews for alternative A_i with respect to feature c_j, $Q_{ij} < n$. The integrated results are shown in Table X.

TABLE X
RESULTS OF INTEGRATED PLTSS

Features	*Products*					
	Samsung			*LG TV*		
Picture quality	0.07	0.29	0.64	0.05	0.20	0.75
Remote control	0.09	0.42	0.49	0.08	0.43	0.49
Sound quality	0.08	0.45	0.47	0.04	0.35	0.61

Next, the geometric averaging operator proposed in [2] is used to determine the comprehensive performance of each product. The compensation level of the larger probabilities to smaller probabilities in s_α is considered by adding a compensation coefficient θ ($\theta > 0$) to the aggregation operator. By Eq. (3), the PLTSs $h_S^{ij}(p) = \{s_\alpha(p_{\alpha,ij}) | \alpha = \{0,1,2\}\}$ of product A_i ($i \in \{1,2\}$) under the three features c_j ($j \in \{1,2,3\}$), are aggregated into a comprehensive PLTS $h_S^i(p) = \{s_\alpha(p_{\alpha,i}) | \alpha = \{0,1,2\}\}$.

$$p_{\alpha,i} = \begin{cases} 0, if\, p_{\alpha,ij} = 0, \forall j \in \{1,2,3\} \\ \dfrac{\prod_j^3 (p_{\alpha,ij} + \theta)^{w_j} - \theta}{\sum_{\alpha=0}^{2} \{\prod_j^3 (p_{\alpha,ij} + \theta)^{w_j} - \theta\}} \quad if \exists j \in \{1,2,3\}, p_{\alpha,ij} > 0 \end{cases} \quad (3)$$

where w_j ($w_j > 0, j \in \{1,2,3\}$) is the weight (importance) of feature c_j, $\sum_{j=1}^{n} w_j = 1$. Here, we assume that the three features are equally important. The compensation coefficient θ is used to control the compensation level between different probabilities. The greater the value of θ is, the larger the degree of compensation is. Without loss of generality, we set θ to 0.1. We can compare the two products according to their comprehensive PLTSs. Let $E(h_S^i(p))$ be the expected value of A_i as defined by Eq. (4) [2]. If $E(h_S^1(p)) > E(h_S^2(p))$, then $h_S^1(p) > h_S^2(p)$, and A_1 is superior to A_2, denoted as $A_1 \succ A_2$; If $E(h_S^1(p)) = E(h_S^2(p))$, then $h_S^1(p) = h_S^2(p)$, and A_1 is indifferent to A_2, denoted as $A_1 \sim A_2$; otherwise, A_1 is inferior to A_2, denoted as $A_1 \prec A_2$.

$$E(h_S^i(p)) = \sum_{\alpha=0}^{2} \frac{\alpha}{2} \times p_\alpha \quad (4)$$

The comprehensive PLTSs of the two products and their utilities are showed in Table XI.

Since $E(h_S^1(p)) < E(h_S^2(p))$, then $h_S^1(p) < h_S^2(p)$, that is, the Samsung product is inferior to the LG TV product. Since we only consider three features and implement sentiment classification by the DL-based model, the ranking result of Samsung and LG TV is different from that determined in [2]. This case study demonstrates how to use the extracted product features and generated PLTSs from the online reviews in our approach to solve online product ranking problems.

TABLE XI
THE COMPREHENSIVE PLTSS AND UTILITIES

Products	*Comprehensive PLTSs*			*Utility*
	Negative	*Neutral*	*Positive*	
Samsung	0.12	0.37	0.51	0.70
LG TV	0.10	0.32	0.58	0.74

VI. CONCLUSION

A. Management Implication

The implementation of effective engineering and technology management (EM&TM) solutions relies on two key elements. Firstly, management processes which combined tools and techniques are needed for supporting management decisions and actions to address specific business problems [37], [64]. Second, conceptual frameworks are needed to guide thinking about technology management, based on well-founded theoretical principles [73]. In this part, we illustrate how our work achieves such requirements to implement effective EM&TM in practice.

For online product text reviews, reviewers quantitatively express their overall satisfaction degrees by star ratings, and qualitatively depict the performance of products/services under different features. It provides a simple and straightforward way for online customers to know the quality of products [1]. However, customers may be confused by several inconsistent reviews caused by different individual preferences, inconsistent product quality, and unreal praises induced by vendors [2]. Limited by the ability of information processing, customers are easily misled by biased reviews and consequently make non-ideal purchase decisions [3]. Our proposed DL-based approach can solve these problems by mining the quantitative and qualitative information from text reviews and translating it into standardized PLTSs. Different from most of the current works mentioned before which simply presented a review being positive/neutral/negative, our approach could quantitatively analyze why this review is positive/neutral/negative and in which product features. Furthermore, we provided a comparative analysis of reviewed literature in Table XII to demonstrate the advantages of our proposed approach. All of these show our work achieves the goal from the first point of effective EM&TM.

Based on existing studies and technologies such as cloud computing and service/event-based architecture in the business platform [73], [74] and our research outcomes, our proposed approach can be smoothly integrated with online shopping platforms to make online customers' shopping process smoother and more efficient. From [75], Jun provided an empirical data analysis to show a strong correlation between searches based on the opinions or recommendations available online and purchase decisions. With our approach, 1) customers could rank products based on their preferences by selecting

different product features; 2) the platform could also have an efficient product recommendation system to better satisfy the need of customers; 3) the vendors could better understand the weakness of their products mentioned in reviews. Furthermore, our approach can bring benefits not only to online shopping but also to other areas. For example, apart from deploying our approach in online shopping platforms, the approach can also be used to build personalized and customized services to support the operations of enterprises/manufacturers depending on the needs in different scenarios. All these greatly improve shopping efficiency and satisfaction, which also show our work achieves the goal from the second point of effective EM&TM.

TABLE XII
COMPARATIVE ANALYSIS OF REVIEWED LITERATURE

Works in online product/service ranking	*Features extraction*	*Feature texts retrieve*	*ML-based method*	*DL-based method*	*Big data processing*
[10], [12]-[16], [27], [28], [30]-[31]	√	×	√	×	×
[20-26]	×	×	√	×	×
[32], [33], [39] – [43]	×	×	×	√	√
Our approach	√	√	√	√	√

B. Discussion and Limitations

Traditional ML-based methods have been used by scholars to extract PLTSs from online product reviews. However, how to use large scale datasets and DL models to improve the PLTS extractions have not yet been well-studied. To overcome the drawbacks of current research, this paper proposed a DL-based sentiment analysis approach to generate PLTSs from online reviews. Based on a large-scale dataset collected from the reviews of online products on Amazon, an algorithm was utilized to produce the training dataset for different categories of products. We also proposed an effective NLP-based method to extract product features from online reviews and retrieve the corresponding sentences that only represent each product feature. In addition, we borrowed the idea of transfer learning which can equip a model with strong prior knowledge before applying the model to a new domain. In our case, we explored various state-of-the-art DL-based methods to build a feature text sentiment classification model of which the prior knowledge was learned from the large-scale training review dataset. Experimental results demonstrate: 1) the methods of matching the rating scores of 1-5 with 5 parts' LTS, and dividing reviews into "positive" and "negative" in existing work cannot train a text classification models to identify correct sentiment tendencies from online reviews; 2) our method of matching the rating scores with the appropriate size of LTS can overcome this problem and benefit the training of a text classification model to identify sentiment tendencies from review texts; 3) our approach achieves high prediction accuracy and competitive performance in the problem of sentiment classification for feature text in online reviews.

In our approach, to fully capture the sentiments conveyed by emojis, we convert the emojis from the input reviews to their corresponding text descriptions for the better product review analysis. However, some of the emojis convey multiple senses of sentiments such as "smile face with tears". Moreover, some emojis may also be used to express more complex semantics such as irony and sarcasm which has a contradictory sentiment with the text [76]. For example, the emoji in text "*It rained heavily today, and I missed the bus :-)*" has a negative sentiment which is opposite with its description texts "*happy face smiley*". All these emoji usages in which the sentiments between the texts and the original meaning of emojis are seriously inconsistent will most probably cause a lot of confusion for our method. This kind of case (expressing negative/positive sentiments with the use of words/emoji with opposite literal meanings) is called sarcasm in linguistics [77]. In current research on NLP, sarcasm identification in text documents from social media data has become an essential research direction and is one of the most challenging tasks due to the lack of advanced embedding models to understand the correct meaning behind sarcasm words/emoji [78]. Therefore, one limitation of our approach is the meaning of emojis in different linguistic and culture backgrounds (aka sarcasm) may not be always identified correctly. Future research may focus on applying more advanced emoji or word embedding models (e.g., Emoji2Vec [79], Topic-enriched word embedding scheme [77], and Inverse gravity moment-based term weighted word embedding model [78]) to understand the semantics behind the emojis and pre-process our review data.

Although we used a large amount of data to train the text classification models in the experiments, scenarios of limited or non-training data for a particular product happen from time to time, e.g.., some unsold or rare products on Amazon. To deal with such situations, in the future, we could consider small data learning methods. In addition, oversampling-duplicating samples from minority classes may be considered to deal with the unbalanced training dataset directly. Lastly but not least, how to improve the performance of sentiment tendency prediction under limited numbers of training reviews for our approach is another research problem worth investigating. We plan to explore meta-learning along this direction of research.

APPENDIX

This article has a supplementary appendix file provided by the authors. Please click the DOI of this paper to access the journal page to download the file.

ACKNOWLEDGMENT

The authors thank editors and anonymous reviewers for their conscientious and constructive comments in the review process.

REFERENCES

[1] Z. P. Fan, G. M. Li, and Y. Liu, "Processes and methods of information fusion for ranking products based on online reviews: An overview," *Information Fusion*, vol. 60, pp. 87–97, Aug. 2020.

[2] X. Wu and H. Liao, "Modeling personalized cognition of customers in online shopping," *Omega (Westport)*, vol. 104, p. 102471, Oct. 2021.

[3] X. Wu and H. Liao, "Learning judgment benchmarks of customers from online reviews," *OR Spectrum*, vol. 43, no. 4, pp. 1125–1157, 2021.

[4] Q. Pang, H. Wang, and Z. Xu, "Probabilistic linguistic term sets in multi-attribute group decision making," *Inf Sci (N Y)*, vol. 369, pp. 128–143, Nov. 2016.

[5] M. Zhao, X. Shen, H. Liao, and M. Cai, "Selecting products through text reviews: An MCDM method incorporating personalized heuristic judgments in the prospect theory," *Fuzzy Optimization and Decision Making*, 2021.

[6] H. gang Peng, H. yu Zhang, and J. qiang Wang, "Cloud decision support model for selecting hotels on TripAdvisor.com with probabilistic linguistic information," *Int J Hosp Manag*, vol. 68, pp. 124–138, Jan. 2018.

[7] P. Liu and F. Teng, "Probabilistic linguistic TODIM method for selecting products through online product reviews," *Inf Sci (N Y)*, vol. 485, pp. 441–455, Jun. 2019.

[8] D. Liang, Z. Dai, M. Wang, and J. Li, "Web celebrity shop assessment and improvement based on online review with probabilistic linguistic term sets by using sentiment analysis and fuzzy cognitive map," *Fuzzy Optim Decis Making*, vol. 19, pp. 561–586, 2020.

[9] G. Kou *et al.*, "A cross-platform market structure analysis method using online product reviews," *Technological and Economic Development of Economy*, vol. 27, no. 5, pp. 992–1018, 2021.

[10] Z. Yan, M. Xing, D. Zhang, and B. Ma, "EXPRS: An extended pagerank method for product feature extraction from online consumer reviews," *Information & Management*, vol. 52, no. 7, pp. 850–858, Nov. 2015.

[11] S. Tirunillai and G. J. Tellis, "Mining Marketing Meaning from Online Chatter: Strategic Brand Analysis of Big Data Using Latent Dirichlet Allocation," *Journal of Marketing Research*, vol. 51, no. 4, pp. 463–479, 2014.

[12] Y. Guo, S. J. Barnes, and Q. Jia, "Mining meaning from online ratings and reviews: Tourist satisfaction analysis using latent dirichlet allocation," *Tour Manag*, vol. 59, pp. 467–483, Apr. 2017.

[13] J. W. Bi, Y. Liu, Z. P. Fan, and J. Zhang, "Wisdom of crowds: Conducting importance-performance analysis (IPA) through online reviews," *Tour Manag*, vol. 70, pp. 460–478, Feb. 2019.

[14] A. Kangale, S. K. Kumar, M. A. Naeem, M. Williams, and M. K. Tiwari, "Mining consumer reviews to generate ratings of different product attributes while producing feature-based review-summary," *Int J Syst Sci*, vol. 47, no. 13, pp. 3272–3286, 2016.

[15] T.-L. Wong and W. Lam, "Hot item mining and summarization from multiple auction web sites," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 2005, pp. 4—pp.

[16] T.-L. Wong and W. Lam, "Learning to extract and summarize hot item features from multiple auction web sites," *Knowl Inf Syst*, vol. 14, no. 2, pp. 143–160, 2008.

[17] J. Cai, J. Li, W. Li, and J. Wang, "Deeplearning Model Used in Text Classification," in *2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2018, pp. 123–126.

[18] R. Socher *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[19] Z. Zhang, Q. Ye, Z. Zhang, and Y. Li, "Sentiment classification of Internet restaurant reviews written in Cantonese," *Expert Syst Appl*, vol. 38, no. 6, pp. 7674–7682, Jun. 2011.

[20] Z. P. Fan, Y. J. Che, and Z. Y. Chen, "Product sales forecasting using online reviews and historical sales data: A method combining the Bass model and sentiment analysis," *J Bus Res*, vol. 74, pp. 90–100, May 2017.

[21] D. Zhang, H. Xu, Z. Su, and Y. Xu, "Chinese comments sentiment classification based on word2vec and SVMperf," *Expert Syst Appl*, vol. 42, no. 4, pp. 1857–1863, Mar. 2015.

[22] F. Tian *et al.*, "A topic sentence-based instance transfer method for imbalanced sentiment classification of Chinese product reviews," *Electron Commer Res Appl*, vol. 16, pp. 66–76, Mar. 2016.

[23] Y. Liu, J.-W. Bi, and Z.-P. Fan, "A Method for Ranking Products Through Online Reviews Based on Sentiment Classification and Interval-Valued Intuitionistic Fuzzy TOPSIS," *Int J Inf Technol Decis Mak*, vol. 16, no. 06, pp. 1497–1522, Sep. 2017.

[24] U. Schmalz, J. Ringbeck, and S. Spinler, "Door-to-door air travel: Exploring trends in corporate reports using text classification models," *Technol Forecast Soc Change*, vol. 170, p. 120865, Sep. 2021.

[25] Y. Hu and W. Li, "Document sentiment classification by exploring description model of topical terms," *Comput Speech Lang*, vol. 25, no. 2, pp. 386–403, Apr. 2011.

[26] N. A. Vidya, M. I. Fanany, and I. Budi, "Twitter Sentiment to Analyze Net Brand Reputation of Mobile Phone Providers," *Procedia Comput Sci*, vol. 72, pp. 519–526, Jan. 2015.

[27] A. Onan, S. Korukoğlu, and H. Bulut, "Ensemble of keyword extraction methods and classifiers in text classification," *Expert Syst Appl*, vol. 57, pp. 232–247, Sep. 2016.

[28] A. Onan, "An ensemble scheme based on language function analysis and feature engineering for text genre classification," *J. Inf. Sci.*, vol. 44, no. 1, pp. 28–47, Feb. 2018.

[29] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting Deep Learning Models for Tabular Data," *arXiv preprint arXiv:2106.11959*, 2021.

[30] A. Onan and S. Korukoğlu, "A feature selection model based on genetic rank aggregation for text sentiment classification," *J. Inf. Sci.*, vol. 43, no. 1, pp. 25–38, Feb. 2017.

[31] M. Al-Smadi, O. Qawasmeh, M. Al-Ayyoub, Y. Jararweh, and B. Gupta, "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews," *J Comput Sci*, vol. 27, pp. 386–393, Jul. 2018.

[32] H. H. Do, P. W. C. Prasad, A. Maag, and A. Alsadoon, "Deep Learning for Aspect-Based Sentiment Analysis: A Comparative Review," *Expert Syst Appl*, vol. 118, pp. 272–299, Mar. 2019.

[33] M. E. Mowlaei, M. Saniee Abadeh, and H. Keshavarz, "Aspect-based sentiment analysis using adaptive aspect-based lexicons," *Expert Syst Appl*, vol. 148, p. 113234, Jun. 2020.

[34] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[36] C. Mühlroth and M. Grottke, "Artificial Intelligence in Innovation: How to Spot Emerging Trends and Technologies," *IEEE Trans Eng Manag*, vol. 69, no. 2, pp. 493–510, 2022.

[37] M. Azzam and R. Beckmann, "How AI Helps to Increase Organizations' Capacity to Manage Complexity - A Research Perspective and Solution Approach Bridging Different Disciplines," *IEEE Trans Eng Manag*, pp. 1–14, 2022.

[38] Y. Lu, X. Xiong, W. Zhang, J. Liu, and R. Zhao, "Research on classification and similarity of patent citation based on deep learning," *Scientometrics*, vol. 123, no. 2, pp. 813–839, 2020.

[39] S. Jiang, J. Hu, C. L. Magee, and J. Luo, "Deep Learning for Technical Document Classification," *IEEE Trans Eng Manag*, pp. 1–17, 2022.

[40] Y. Wang, L. Luo, and H. Liu, "Bridging the Semantic Gap Between Customer Needs and Design Specifications Using User-Generated Content," *IEEE Trans Eng Manag*, vol. 69, no. 4, pp. 1622–1634, 2022.

[41] A. Onan, "Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks," *Concurr Comput*, vol. 33, no. 23, p. e5909, 2021.

[42] A. Onan, "Mining opinions from instructor evaluation reviews: A deep learning approach," *Computer Applications in Engineering Education*, vol. 28, no. 1, pp. 117–138, 2020.

[43] A. Onan, "Sentiment analysis on massive open online course evaluations: A text mining and deep learning approach," *Computer Applications in Engineering Education*, vol. 29, no. 3, pp. 572–589, 2021.

[44] C. C. Li, Y. Dong, F. Herrera, E. Herrera-Viedma, and L. Martínez, "Personalized individual semantics in computing with words for supporting linguistic group decision making. An application on consensus reaching," *Information Fusion*, vol. 33, pp. 29–40, Jan. 2017.

[45] A. Onan, S. Korukoğlu, and H. Bulut, "A hybrid ensemble pruning approach based on consensus clustering and multi-objective

evolutionary algorithm for sentiment classification," *Inf Process Manag*, vol. 53, no. 4, pp. 814–833, Jul. 2017.
[46] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in *Proceedings of the 2014 conference on empirical methods in natural language processing*, 2014, pp. 1746–1751.
[47] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *Adv Neural Inf Process Syst*, vol. 28, pp. 649–657, 2015.
[48] P. Liu, X. Qiu, and X. Huang, "Recurrent Neural Network for Text Classification with Multi-Task Learning," in *IJCAI*, 2016.
[49] I. Sutskever, O. Vinyals, and Q. v Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.
[50] T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," in *EMNLP*, 2015.
[51] C. Du and L. Huang, "Text Classification Research with Attention-based Recurrent Neural Networks," *INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL*, vol. 13, no. 1, pp. 50–61, 2018.
[52] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
[53] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *NAACL-HLT (1)*, 2019, pp. 4171–4186.
[54] Z. Shaheen, G. Wohlgenannt, and E. Filtz, "Large scale legal text classification using transformer models," *arXiv preprint arXiv:2010.12871*, 2020.
[55] Y. Zhu *et al.*, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19–27.
[56] A. Joulin, É. Grave, P. Bojanowski, and T. Mikolov, "Bag of Tricks for Efficient Text Classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017, pp. 427–431.
[57] T. Yao, Z. Zhai, and B. Gao, "Text Classification Model Based on fastText," in *2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS)*, 2020, pp. 154–157.
[58] S. Gao *et al.*, "Limitations of Transformers on Clinical Text Classification," *IEEE J Biomed Health Inform*, vol. 25, no. 9, pp. 3596–3607, 2021.
[59] B. Rodrawangpai and W. Daungjaiboon, "Improving text classification with transformers and layer normalization," *Machine Learning with Applications*, vol. 10, p. 100403, 2022.
[60] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.
[61] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 188–197.
[62] M. Fernández-Gavilanes, E. Costa-Montenegro, S. García-Méndez, F. J. González-Castaño, and J. Juncal-Martínez, "Evaluation of online emoji description resources for sentiment analysis purposes," *Expert Syst Appl*, vol. 184, p. 115279, Dec. 2021
[63] C. J. van Rijsbergen, S. E. Robertson, and M. F. Porter, *New models in probabilistic information retrieval*, vol. 5587. British Library Research and Development Department London, 1980.
[64] S. Kheybari, F. Mahdi Rezaie, and J. Rezaei, "Measuring the Importance of Decision-Making Criteria in Biofuel Production Technology Selection," *IEEE Trans Eng Manag*, vol. 68, no. 2, pp. 483–497, 2021.
[65] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans Knowl Data Eng*, vol. 22, no. 10, pp. 1345–1359, 2009.
[66] T. Semwal, P. Yenigalla, G. Mathur, and S. B. Nair, "A Practitioners' Guide to Transfer Learning for Text Classification using Convolutional Neural Networks," in *Proceedings of the 2018 SIAM International Conference on Data Mining (SDM)*, pp. 513–521.
[67] Z. Liu, Q. Wang, and F. Meng, "A benchmark for multi-class object counting and size estimation using deep convolutional neural networks," *Eng Appl Artif Intell*, vol. 116, p. 105449, 2022.
[68] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.
[69] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Adv Neural Inf Process Syst*, vol. 32, 2019.
[70] L. Wang, M. Han, X. Li, N. Zhang, and H. Cheng, "Review of classification methods on unbalanced data sets," *IEEE Access*, vol. 9, pp. 64606–64628, 2021.
[71] A. Kumar, S. Goel, et al. "A Review on Unbalanced Data Classification," in *Proceedings of International Joint Conference on Advances in Computational Intelligence*, P. K. and B. J. C. Uddin Mohammad Shorif and Jamwal, Ed., Singapore: Springer Nature Singapore, 2022, pp. 197–208.
[72] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.
[73] S. Ozcan, A. Homayounfard, C. Simms, and J. Wasim, "Technology Roadmapping Using Text Mining: A Foresight Study for the Retail Industry," *IEEE Trans Eng Manag*, vol. 69, no. 1, pp. 228–244, 2022.
[74] Z. Liu *et al.*, "The architectural design and implementation of a digital platform for Industry 4.0 SME collaboration," *Comput Ind*, vol. 138, p. 103623, Jun. 2022.
[75] S. P. Jun and D. H. Park, "Consumer information search behavior and purchasing decisions: Empirical evidence from Korea," *Technol Forecast Soc Change*, vol. 107, pp. 97–111, Jun. 2016.
[76] Y. Chen, J. Yuan, Q. You, and J. Luo, "Twitter sentiment analysis via bi-sense emoji embedding and attention-based LSTM," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 117–125.
[77] A. Onan, "Topic-Enriched word embeddings for sarcasm identification," in *Software Engineering Methods in Intelligent Algorithms*, 2019, pp. 293–304.
[78] A. Onan and MA. Toçoğlu, "A term weighted neural language model and stacked bidirectional LSTM based framework for sarcasm identification." IEEE Access, vol. 9, pp. 7701-7722, 2021.
[79] B. Eisner, T. Rocktäschel, I. Augenstein, M. Bosnjak, and S. Riedel, "emoji2vec: learning emoji representations from their description," in *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*, 2016, pp. 48–54.

Zixu Liu received his PhD in Computer Science at The University of Manchester, Manchester, United Kingdom, in 2018, his MSc in Computation and Game Theory at the University of Liverpool, Liverpool, United Kingdom, in 2013 and his BSc in Computer Science and Technology at Jilin University, Changchun, China, in 2011.

Dr. Liu is currently a Lecturer in Decision Analytics and Risk at Southampton Business School, University of Southampton, Southampton, United Kingdom. His main research areas are machine learning, information system, and decision-making. He mainly uses state-of-the-art computer technics from machine learning, and optimization to solve the business analytics problems such as multicriteria decision-making problems, electricity market dynamic pricing, and object counting and size estimation problems. Also, he utilizes his programming skills and knowledge of soft engineering to transfer his research and knowledge to Industry.

Huchang Liao (M'13–SM'17) is a Research Fellow at the Business School, Sichuan University, Chengdu, China. He received his PhD degree in Management Science and Engineering from the Shanghai Jiao Tong University, Shanghai, China, in 2015. He has published 4 monographs, and more than 320 peer-

reviewed papers, many in high-quality international journals including *Decision Analysis*, *European Journal of Operational Research*, *Omega, IEEE Transactions on Fuzzy Systems, IEEE Transactions on Engineering Management, IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *IEEE Transaction on Cybernetics,* etc. His publications have been cited over 16,000 times, and his h-index is 70. He has been a Highly Cited Researcher in Computer Science (2019-2022), and a Highly Cited Chinese Researchers in Management Science (2020-2022). He ranked within the top 2% Ranking of Scientists in the World by Stanford University (2020-2022). His main research interests include multiple criteria decision analysis, group decision analysis, fuzzy decision analysis, machine learning based decision analysis and medical decision analysis. Prof. Liao has been elected to be the Fellow of IET, the Fellow of BCS, the Fellow of RSA, and the Fellow of IAAM. He is the Area Editor of *International Journal of Information Technology & Decision Making* (SCI), Associate Editor, Guest Editor or Editorial Board Member for many top-tier international journals, including *IEEE Transactions on Fuzzy Systems*, *Information Fusion*, *Applied Soft Computing*, *Omega*, *Technological and Economic Development of Economy*, *International Journal of Strategic Property Management*, *Engineering Applications of Artificial Intelligence*, and *International Journal of Fuzzy Systems*.

Maolin Li received the Ph.D. degree in computer science from the University of Manchester, Manchester, United Kingdom, in 2021. His research interests include natural language processing and text mining, with a focus on information extraction with probabilistic and deep learning models.

Qian Yang is now pursuing her master's degree in management science at Sichuan University, Sichuan, China. Her current research interests include multiple attribute decision making, preference learning.

Fanlin Meng received his PhD in Computer Science from the University of Manchester in 2015 where he is currently a Lecturer in Data Science. He received the BSc in Automation from China University of Mining and Technology, China in 2008, and the MSc in Systems Engineering from Xiamen University, China in 2011. He is Fellow of British Computer Society (FBCS), member of EPSRC Peer Review College, and Senior Member of IEEE. His primary research interests include Energy Market, Smart Energy and Mobility, Operations Research, Machine Learning, Game Theory and Optimization.