IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 22 September 2023, accepted 6 October 2023, date of publication 16 October 2023, date of current version 20 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3324654

RESEARCH ARTICLE

Efficient Detection of Noise Reviews Over a Large Number of Places

HYEON GYU KIM[1] **AND YOO HYUN PARK**[2]
[1]Department of Computer Science and Engineering, Sahmyook University, Seoul 01795, Republic of Korea
[2]Department of Computer Software Engineering, Dong-Eui University, Busan 47340, Republic of Korea

Corresponding author: Yoo Hyun Park (yhpark@deu.ac.kr)

ABSTRACT User reviews have been widely used to extract opinions, complaints, and requirements about a given place or product from the users' point of view. In the process of collecting user reviews, a lot of reviews irrelevant to a given search keyword are included in the collection result. Such irrelevant reviews can easily be detected using supervised learning algorithms. However, situation changes when the number of places or products that need to be analyzed increases because manual labeling is required for each collected review. This paper presents a method to detect irrelevant reviews efficiently when a large number of places and reviews need to be analyzed. The basic idea of the proposed method is to expand the target of the learning from an individual place to a group consisting of multiple places. The method can be applied properly to any place whose number of reviews is not enough to perform the training because a classifier obtained by training sufficient reviews included in the places of a group can be used for the places with insufficient reviews in the group. As an initial study of this approach, we tried to check through experiments whether the proposed method can provide higher accuracy than the conventional method where the training is performed for individual places. Our experimental results showed that the average f1-score of the group learning was about 0.931 over real data collected online. For the places with less than 100 reviews, the f1-score of the conventional and proposed methods was 0.651 and 0.865, respectively, showing that 21.4% performance improvement.

INDEX TERMS Spam review detection, noise reviews, group learning, LSTM, BERT, DistilBERT.

I. INTRODUCTION

With the popularization of the Internet, sharing reviews of places or products through SNS platforms such as Facebook or Twitter has become commonplace. Since these reviews contain users' opinions, complaints, and requirements, various attempts have been made to extract values from the reviews [1], [2]. Nowadays, there is a growing interest in applications that extract and utilize user opinions comprehensively from a large number of Places of Interest (POIs) [3] such as restaurants, hotels, and tourist attractions. Typical examples of the applications are as follows.

- *Travel route recommendation* [4], [5]: Finds popular restaurants, hotels and tourist attractions in a specific region, and recommends the shortest route connecting them.
- *Nearby place recommendation* [6]: Finds popular places located within a radius of the predefined N meters from the user's current location, then sorts and shows them in order of their popularity.
- *Regional commercial status analysis* [7]: For the places such as restaurants, cafes, hotels, etc. located on a specific region, analyzes and shows their distributions, reputations, and competitors.

To implement these applications, it is necessary to collect user reviews for each place. For this purpose, open search APIs provided by online platforms such as Twitter or Google are popularly used. However, their results may include a number of reviews that are not related to a given search term. For example, when "*Hamyang House*", a famous restaurant in the Ulsan city, South Korea, is given as a

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wang.

search term, the results may include the irrelevant reviews as follows.

- *Romantic Table*, a fine dining place near *Hamyang House* in Sinjung-dong, Namgu, Ulsan ...
- Seasonal special discount at *Jun Hair* located on the 2nd floor of the *Hamyang House* building ...

In these reviews, a search term is simply being borrowed to describe a place other than the one specified by the search term. The majority of the irrelevant reviews have such a form. Other forms of reviews can also be seen in the search results. For example, some reviews have text randomly generated from the search term. Some reviews only have a list of names without detail information, such as reviews introducing a bus route where the names of bus stops are simply listed. For simplicity, these reviews are referred to as *noise reviews* in the sense that they can distort analysis results when included. Other reviews that are not noisy are referred to as *legitimate reviews* in this paper.

The problem is that these noise reviews account for a too high percentage of search results. An average of 55.4% of the reviews for restaurants and 73.5% of the reviews for tourist attractions were found noisy (see Chapter 3). Hence, in order to obtain accurate results in the recommendation services mentioned above, it is necessary to detect and filter out these reviews prior to the analysis.

Noise reviews can be viewed as a sub-type of *spam reviews* discussed in literature. Jindal and Liu [8], [9] first introduced the concept of spam reviews and described three types of spam reviews as follows.

- *Type 1* (*untrustful opinions*): Those that deliberately mislead readers by providing undeserving positive reviews to target objects in order to promote the objects or by giving malicious negative reviews in order to damage their reputation.
- *Type 2* (*reviews on brands only*): Those that do not provide any useful information on the target products in reviews, but only the brands, the manufacturers or the sellers of the products. These reviews are considered as spam because they are not targeted at specific products.
- *Type 3* (*non-reviews*): Those that are non-reviews, which have two main sub-types: (1) advertisement and (2) other irrelevant reviews containing no opinions.

From the above point of view, noise reviews can be defined as *irrelevant reviews containing opinions on another objects, not the target objects, as well as without opinions*, and are included in Type 3. Jindal and Liu discussed that Type 2 and 3 reviews are easier to identify than Type 1, and can easily be isolated using machine learning. From the discussion, most follow-up studies have focused on the detection of Type 1 reviews [10], [11], [12], [13]. Regarding Type 2 and 3, studies have not been sufficiently conducted; simple, old-fashioned machine learning algorithms were tested with the small size of data sets. For example, Jindal and Liu [9] simply run logistic regression on the data using only 470 reviews for the detection of those reviews. Raymond et al. [14] discussed that SVM (Support Vector Machine) showed better performance compared to K-Nearest Neighbor and logistic regression, but their test data sets were synthesized, not the real reviews.

Note that the noise reviews can easily be seen nowadays as social big data analysis has been actively conducted in various domains. Many applications such as POI recommendation and sentiment analysis require the analysis of the large amount of user reviews and use open search APIs to collect reviews [15], [16], [17]. Naturally, in these applications, it is essential to filter out noise reviews in advance from the results returned from the search APIs. The detection of Type 1 or 2 reviews can be performed as a separate process after the filtering.

It is not so difficult to detect noise reviews using machine learning algorithms, as discussed by Jindal and Liu. But this holds only when the number of places and their reviews is relatively small. If the number becomes large, the problem becomes harder. For example, suppose that we want to build a travel route recommendation service for the places in Ulsan. To build the service, we need to collect user reviews for the places and filter out noise reviews for each place. However, the number of restaurants and tourist attractions registered in the city was about 16,900 and 150, respectively. Moreover, up to 1,000 reviews can be collected for each place if we use open search APIs provided by Naver [18], a popular online portal service of Korea. In this case, to analyze all the places in the city, it is necessary to arithmetically process up to 17,050,000 reviews. Manual labeling of all these reviews to conduct a supervised learning is a completely exhausted job.

Another problem is that many places in the city were found to have insufficient reviews. In general, when the number of training data is less than 100, it is difficult to obtain meaningful accuracy in the supervised learning. Unfortunately, more than 60% of restaurants in Ulsan have less than 100 reviews (see Chapter 3). If those places are excluded from the analysis, accuracy of the recommendation service can significantly be degraded. In case of the small rural towns, the problem becomes worse as the number of reviews collected per place is even smaller.

This paper presents *group learning* to resolve the problems of the noise review detection when a large number of places and reviews need to be analyzed. The basic idea of the proposed method is to expand the target of supervised learning from an individual place to a *group* consisting of multiple places. In this approach, as the number of places in a group increases, the number of groups to be learned decreases, which leads to increasing efficiency. Also, as the characteristics of the places in a group are similar, higher accuracy can be achieved. In this way, the efficiency and accuracy of learning can be closely affected by how places are grouped in this approach. For simplicity, the conventional approach that performs the learning for each place is referred to as *individual learning* below.

To the best of our knowledge, no studies have discussed how to perform group learning for the problem of noise

review detection. Regarding this, the following four aspects of the group learning are discussed in this paper.

① *Grouping criterion*: a criterion for grouping places should be set properly to achieve satisfactory accuracy and efficiency.
② *Data sampling for supervised learning*: the size of a data set of a group to conduct supervised learning should be kept small enough to be manually labeled, regardless of how many reviews are included in the group. Also, the characteristics of the original data should be reflected properly to the sampled data.
③ *Detection performance*: accuracy of the group learning should be similar or better than that of the individual learning in the detection of noise reviews.
④ *Performance with insufficient data*: The group learning should be applied properly for the places where training is difficult due to the lack of reviews.

As an initial study, this paper focuses on verifying the feasibility of the proposed group learning through experiments using the real data; theoretical verification will be discussed in future research. To simplify the discussion, the criteria related to ① and ② were determined heuristically to get the simplest form of the group learning that enables manual labeling of the training and test data sets. Experiments were then conducted to check whether the group learning satisfies ③ and ④ on the real data. About 1.6 million reviews for the 17,050 places in Ulsan were collected and used for the experiments. To develop a model for the detection, LSTM (Long Short-Term Memory) [19], BERT (Bidirectional Encoder Representation for Transformers) [20] and Distil-BERT (a distilled version of BERT) [21] were adopted, which are known to provide high accuracy in text processing.

The remaining part of this paper is organized as follows. Section II introduces existing studies to detect spam reviews using machine learning. Section III discusses how to construct experimental data sets along with the assumptions on heuristic criteria related to ① and ②, and explains how to implement a model for the noise review detection using LSTM, BERT, and DistilBERT. Section IV compares performance of the three models over the experimental data, and discusses the results in terms of ③ and ④. Section V concludes the paper with future research directions regarding ① and ②.

II. RELATED WORK

As discussed in [10], [11], and [22], the majority of the existing studies have focused on the detection of Type 1 reviews. This section discusses the studies in terms of their machine learning algorithms and performance achievement. The studies for the detection of Type 2 and 3 reviews are also briefly discussed.

A. TRADITIONAL MODELS

Existing studies using traditional machine learning algorithms, such as Logistic Regression (LR), Naïve Bayes (NB), Random Forest (RF), and Support Vector Machine (SVM), are well summarized in [10]. In this approach, it is important how to extract the characteristics of input data and encode them into discrete features that will be used as inputs for learning. The features used for the detection of spam reviews can be divided into *review-centric features* and *reviewer-centric features* [9]. The former includes features to represent the textual contents of a review, such as n-grams, LIWC (Linguistic Inquiry and Word Count), and POS (Parts of Speech). The latter includes features to represent the behavior of a reviewer who wrote the reviews, including the maximum number of reviews (written by the reviewer), the percentage of positive reviews, review length, and others.

Among the studies, some achieved high accuracy only with the review-centric features. Ott et al. [23] extracted LIWC and bigrams from the hotel reviews obtained through Amazon Mechanical Turk (AMT), a crowdfunding site, and achieved 89.8% accuracy by applying SVM to the features. Their follow-up study [24] suggested that higher accuracy could be achieved by dealing with the positive and negative sentiment reviews separately. Shojaee et al. [25] adopted the *stylometric* feature [26] for high accuracy which is a mixture of lexical and syntactic features of the reviews. Their SVM model achieved 84% f1-score from the hotel reviews presented by Ott et al. Feng et al. [27] extracted Context Free Grammar (CFG) parse trees from the hotel reviews of Ott et al. and obtained 91.2% accuracy using SVM with the features. Li et al. [28] created a new data set by adding reviews generated by domain experts in the field of restaurants and hospitals to the hotel reviews of Ott et al, and extracted unigrams, LIWC, POS from the data. They used the Sparse Additive Generative Model (SAGE), a generative Bayesian approach introduced by Eisenstein et al. [29], and achieved 65% accuracy from the data. Their results showed that it is difficult to obtain high accuracy when the domains of spam reviews become diverse.

Many other studies have employed both of the review and reviewer-centric features for higher accuracy. Jindal and Liu [9] collected 5.8 million reviews from the Amazon website and extracted candidates of spam reviews based on the insight that duplicate or similar reviews are highly likely to be spam, which was also pointed out by Qian and Liu [30]. They achieved 78% AUC using logistic regression with 36 review and reviewer-centric features extracted from the candidates. Li et al. [31] extracted review sentiments (positive or negative) and reviewer's behaviors from 6,000 reviews collected from Epinions, and achieved 63.1% f1-score using Naïve Bayes. Mukherjee et al. [32], [33] suggested that higher accuracy can be obtained using the reviewer's abnormal behavioral features together with the linguistic features, and their SVM model showed 86.1% accuracy for the Yelp data. Li et al. [34] tried to increase accuracy by using reviewer's temporal and spatial patterns, and obtained 85% f1-score using SVM for the Dian-ping data.

B. NEURAL NETWORK MODELS

A neural network is known to have great non-linear fitting capabilities [11]. Its advanced model with well-trained word embeddings can effectively capture the syntactic structures of a text as sell as semantic relationships between the words in a more scalable way. From these characteristics, using deep learning algorithms based on the neural network architecture, it is possible to achieve high accuracy only by analyzing the review texts without analyzing the behavior of reviewers. Also, recent studies on the spam review detection tend to use deep learning nowadays.

Barushka and Hajec [35] achieved 89% accuracy using DNN (Deep Neural Network), the basic neural network, from the hotel reviews of Ott et al. Li et al. [36] employed CNN (Convolutional Neural Network) [37] for the data sets of [28], and obtained 82.3% f1-score. Zhao et al. [38] also used CNN where the order of words appearing in the review is embedded in its convolutional and pooling layers to make it more suitable for detecting short spam reviews. Their algorithm provided 82.8% f1-score for the 24,166 hotel reviews collected online. Shahariar et. al [39] compared the performance of CNN and LSTM, and showed that the LSTM model provided better performance whose f1-score was 94.57% for the hotel reviews of iOtt et al. Wang et al. [40] compared the performance of SVM and LSTM, and also showed that LSTM is superior. Liu et al. [41] used Bi-LSTM with feature combination and achieved 87.6% f1-score for the data sets of [28].

Various attempts have also been made to improve accuracy by combining two or more deep learning algorithms. Wang et al. [42] tried to improve performance by learning linguistic features and behavioral features separately. In their approach, CNN was used to express linguistic features and Attention [43] was used to express relationship between the two features. Their algorithm achieved 88.9% and 91.2% f1-scores for hotel and restaurant reviews in the Yelp data of [32], respectively. Ren and Zhang [44] combined CNN and bi-directional gated RNN (Recurrent Neural Network) [45] to model discourse information and build a document vector, and they achieved 83.9% f1-score for the data sets of [28]. Bhuvaneshwari et al. [46] discussed a similar approach where LSTM, not RNN, was used to build a document vector, and achieved 87% f1-score for the YelpZip data. Bathla et al. [47] used CNN and LSTM for the spam review detection, which is similar to [46], but aimed to reduce computation time and improve accuracy by extracting and using only some aspects of the reviews rather than using the given entire documents for learning.

Recently, Kanmani et. Al. [48] compared the performance of various Transformer models such as BERT [20], Roberta (Robustly Optimized BERT) [49], XLNET (Transformer-XL) [50], and XLM-Roberta (Cross-lingual Language model – Roberta) [51] in the spam review detection. Duma et al. [52] also combined Transformers as well as CNN and LSTM to build vectors from the reviews, and learned the vectors with the aspect and overall ratings information. Using the hybrid model, they achieved 96.5% f1-score.

TABLE 1. Notations and symbols used in this paper.

Symbol	Description
P	A set of places collected online, $P = \{ p_i \}$ $(1 \le i \le N)$
N	Number of places in P
R_i	A set of reviews of the i-th place, p_i
R	A set including reviews of all R_i $(1 \le i \le N)$
C_b	A partition of P consisting of places with a review size between $\lfloor b/2 \rfloor_{50}$ and b where $b \in \{50, 100, 250, 500, 1000\}$ (refer to (1))
$G^{(i)}$	The i-th group of places in P
$C_b^{(i)}$	A partition of $G^{(i)}$ consisting of places with a review size between $\lfloor b/2 \rfloor_{50}$ and b where $b \in \{50, 100, 250, 500, 1000\}$
$\gamma(c)$	A function that returns the number of total reviews of places in c (refer to (2))
$\rho(c)$	A function that returns the percentage of $\gamma(c)$ in total reviews in R (refer to (3))
$\mu(c)$	A function that returns the average number of reviews of places in c $(= \gamma(c)/\lvert c \rvert)$
S	A set of places sampled from P for experiments
$S^{(i)}$	A set of places sampled from $G^{(i)}$ for experiments
$S_b^{(i)}$	A partition of $S^{(i)}$ consisting of places with a review size between $\lfloor b/2 \rfloor_{50}$ and b where $b \in \{50, 100, 250, 500, 1000\}$
M	Target number of reviews that needs to be included in S^i
$\nu(s)$	A function that returns the average number of reviews in the data set s
φ_i	Selection priority of a place p_i

C. DETECTION OF TYPE 2 AND 3 REVIEWS

He et al. [12] discussed that Type 2 and 3 reviews also attract many researchers' interests because these types of reviews have the potential of fraud. Nevertheless, it was difficult to find relevant studies, except the two early studies [9] and [14] mentioned in Chapter I. Jindal and Liu [9] obtained 98.7% AUC using the logistic regression for total 470 Type 2 and 3 reviews. Raymond et al. [14] achieved 95.06% AUC using SVM for 1,032 synthesized reviews. Although they achieved high accuracy, it is difficult to say that the characteristics of the real data are sufficiently reflected to their results. As the use of open search APIs becomes more popular, the detection of Type 3 reviews, including the noise reviews, becomes more important. Due to the increasing demand, it is expected that more research in this area will be required.

III. EXPERIMENTAL SETTINGS

This section describes data and algorithm settings necessary to conduct experiments to show the feasibility of the proposed group learning, including how to collect places and reviews, how to sample and construct data sets necessary to perform supervised learning, and how to implement detection models using LSTM, BERT, and DistilBERT. The notations and symbols used in this paper are summarized in Table 1.

A. DATA PREPARATION

In this paper, experiments were conducted for the places registered in the Ulsan Metropolitan City, which consists of

TABLE 2. Number of places included in the experimental data sets $S_b^{(i)}$ for group $G^{(i)}$ in Ulsan generated by Algorithm 1.

	Namgu		Dongu		Bukgu		Ulju		Joonggu	
	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots
	$G^{(1)}$	$G^{(2)}$	$G^{(3)}$	$G^{(4)}$	$G^{(5)}$	$G^{(6)}$	$G^{(7)}$	$G^{(8)}$	$G^{(9)}$	$G^{(10)}$
$S_{50}^{(i)}$	7	0	8	0	9	0	24	1	7	0
$S_{100}^{(i)}$	2	0	3	0	2	0	3	2	2	0
$S_{250}^{(i)}$	4	0	3	1	3	1	5	0	3	2
$S_{500}^{(i)}$	2	2	1	1	2	1	0	0	3	1
$S_{1000}^{(i)}$	5	6	4	8	5	9	2	5	5	8
$S^{(i)}$	20	8	19	10	21	11	34	8	20	11

5 regional districts including Namgu, Dongu, Bukgu, Ulju, and Joonggu. Ulsan was selected for the experiments because it is the smallest among the seven metropolitan cities in South Korea and the size of the data that needs to be collected for the experiments is relatively small. To simplify the discussion, among the various types of places, only restaurants and tourist attractions were used for experiments. Restaurant information was obtained from the open data provided by Small Business Promotion Agency (SBPA) [53], and about 16,900 restaurants were collected. Tourist attractions were obtained using Tour APIs provided by Korea Tourism Organization [54], and about 150 places were gathered.

Among the collected place data, many errors were found in the restaurants, since changes due to closure or relocation were not properly reflected in the data. To fix the errors, the address of each place was checked to see its validity using the place search API of Naver [18]. For a given place, if a new address returned by the API matched the address specified in the open data of SBPA, the place was considered valid. Using this process, we found 7,561 valid places. For each valid place, user reviews were then collected using the blog search API of Naver [55]. Review collection was conducted in January 2023, and the total number of user reviews collected for the valid places was about 1.6 million.

To check the distribution of the collected data, the places were grouped according to the number of their reviews. Let the set of collected places be P. Then P can be partitioned into 5 classes based on the number of reviews whose values are 1000, 500, 250, 100, and 50. More specifically, let the i-th place of P be p_i and the set of reviews of p_i be R_i. Then, p_i can be assigned to a class C_b according to the number of reviews, $|R_i|$, as follows

$$p_i \in C_b \quad \text{if } \lfloor b/2 \rfloor_{50} < |R_i| \leq b \tag{1}$$

where $b \in \{50, 100, 250, 500, 1000\}$ and $\lfloor x \rfloor_{50} = x - x \ mod \ 50$. $\lfloor x \rfloor_{50}$ denotes the largest multiple of 50 that does not exceed x. For example, if b is given to 250, $\lfloor 250/2 \rfloor_{50}$ is 100. Thus, C_{250} denotes a set of places whose number of reviews is less than or equal to 250 and greater than 100. The maximum value of the bound b was set to 1,000 because when using the Naver search APIs, the maximum number of reviews that can be collected per month for each place was 1,000.

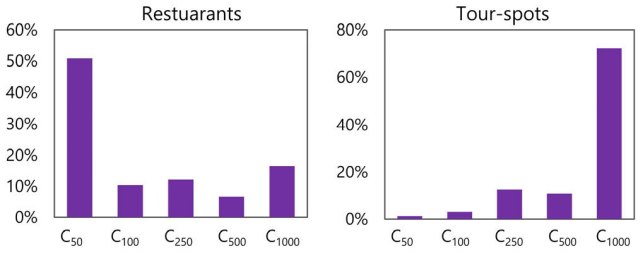

FIGURE 1. Size of C_b in percentage: (a) restaurants (left) and (b) tourist attractions (right).

Figure 1 (a) shows the size of C_b in percentage for the restaurants in Ulsan. The size of C_{50} was the largest and the size of C_{1000} was the second largest, showing a polarized distribution. Note that the combined ratio of C_{50} and C_{100} exceeds 60%. This means that more than half of the restaurants have less than 100 reviews. Thus, if the noise review detection for these places is not performed properly, the accuracy of the services mentioned in Section I can be significantly degraded. Figure 1 (b) shows the size of C_b in percentage for the tourist attractions in Ulsan. The size of C_{1000} was the largest, accounting for 72% of the total. The combined ratio of C_{250}, C_{500}, and C_{1000} exceeds 95%, which means that most tourist attractions have more than 100 reviews.

The size of reviews for each C_b was then compared. The size of reviews of C_b, denoted $\gamma(C_b)$, can be calculated as the sum of the number of reviews of all places belonging to C_b.

$$\gamma(C_b) = \sum |R_i| \quad for \ all \ p_i \in C_b \tag{2}$$

Using $\gamma(C_b)$, the percentage of reviews of C_b, denoted $\rho(C_b)$, can be calculated as follows.

$$\rho(C_b) = \gamma(C_b) / \sum\nolimits_{1 \leq i \leq N} |R_i| \tag{3}$$

Figure 2 (a) shows $\rho(C_b)$ for the restaurants in Ulsan. The ratio of C_{1000} was the largest, accounting for more than 70% of the total. On the other hand, the combined ratio of reviews for C_{50} and C_{100} was very small at 6.5%. This is in contrast to the results of Figure 1 (a) where the combined ratio of C_{50} and C_{100} exceeds 60%. Figure 2 (b) shows $\rho(C_b)$ for tourist attractions in Ulsan. The ratio of C_{1000} was the largest,

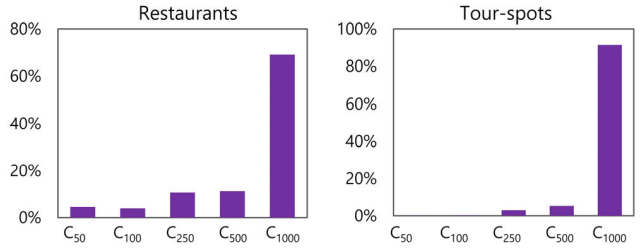

FIGURE 2. Size of reviews of C_b in percentage: (a) restaurants (left) and (b) tourist attractions (right).

accounting for 91% of the total, while the combined ratio of C_{50} and C_{100} was very small, accounting for 0.2%.

As summarized, more than 60% of the restaurants had less than 100 reviews, whereas the portion was about 6.5% of the total reviews. These places can have difficulty in supervised learning due to the lack of reviews. On the other hand, in case of tourist attractions, more than 95% of the places and their reviews are concentrated on C_{250} and above. Thus, training on them can be conducted relatively easy and higher accuracy can be achieved, compared to the case of restaurants.

B. DATA CONSTRUCTION

The experimental data set S can be constructed by sampling some portion of data in P. To conduct the group learning, S is built for each group. Regarding this, the criterion for grouping places was determined heuristically, as mentioned in Section I ①. We assumed that the closer the places are, the more likely it is that their reviews will use common words, such as famous place names, street names, or descriptions of environmental characteristics. The same assumption can be applied to the place categories; as business categories of the places become more similar, the number of common words in their reviews may increase.

From the assumptions, places were grouped according to their locations and categories. To simplify the discussion, a coarse-grained criterion was employed: districts for the location, and restaurants and tourist attractions for the category. As a result, places were divided into 10 groups, including restaurant and tourist attraction groups for each district in Ulsan. More fine-grained criteria can be used for the place grouping to achieve higher accuracy, such as streets for the location, and Korean, Japanese, Chinese cuisines for the category. The relationship between the grouping criteria and detection accuracy is beyond the scope of this paper and will be studied in future research.

Let the i-th group be $G^{(i)}$. The index i can be calculated as $d \times 2 + c + 1$, where d is the index of districts (Namgu, Dongu, Bukgu, Ulju, Joonggu) and c is the index of the categories (restaurants, tour attractions) of the places. For example, the restaurant and tour attraction groups in Dongu can be denoted as $G^{(3)}$ and $G^{(4)}$, respectively.

For each $G^{(i)}$, the experimental data set $S^{(i)}$ is constructed. Note that the size of $S^{(i)}$ should be small enough to be manually labeled for supervised learning. Regarding this, the criterion for the size of a data set was determined heuristically, which was mentioned in Section I ②. In the proposed method, the number of reviews included in $S^{(i)}$ was set to 5,000 which is an empirical size that enables manual labeling. The target size of $S^{(i)}$ is also denoted M, below.

To build $S^{(i)}$ from $G^{(i)}$, the distribution of the number of reviews in C_b was considered. Let $C_b^{(i)}$ denote C_b of $G^{(i)}$, and $\rho(C_b^{(i)})$ denote the percentage of reviews of $C_b^{(i)}$. Then, $S^{(i)}$ can be constructed as follows. First, $S^{(i)}$ is divided into 5 partitions, denoted $S_b^{(i)}$, where $b \in \{50, 100, 250, 500, 1000\}$. $S_b^{(i)}$ is configured to include reviews of places belonging to $C_b^{(i)}$, and the size of $S_b^{(i)}$ is calculated as $\rho(C_b^{(i)}) \times M$. For example, the percentage of reviews of $C_{50}^{(1)}$, $C_{100}^{(1)}$, $C_{250}^{(1)}$, $C_{500}^{(1)}$, and $C_{1000}^{(1)}$ for the group $G^{(1)}$, the restaurants in Ulsan Namgu, was 3.1%, 3.2%, 9.1%, 10.8%, and 73.8%, respectively. In this case, 156, 162, 455, 540, and 3,688 reviews can be included in $S_{50}^{(1)}$, $S_{100}^{(1)}$, $S_{250}^{(1)}$, $S_{500}^{(1)}$, and $S_{1000}^{(1)}$, respectively, for the restaurants.

When the size of $S_b^{(i)}$ is determined, the average number of places that can be included in $S_b^{(i)}$, denoted $\nu(S_b^{(i)})$, can be calculated. Let the average number of places in $C_b^{(i)}$ be $\mu(C_b^{(i)})$, that is, $\gamma(C_b^{(i)}) / |C_b^{(i)}|$. Then, $\nu(C_b^{(i)})$ can be calculated as $|S_b^{(i)}| \ / \ \mu(C_b^{(i)})$. For example, the sum of the number of reviews of $C_{50}^{(1)}$, $\gamma(C_{50}^{(1)})$, was 18,715, and the number of restaurants $|C_{50}^{(1)}|$ was 1,075. In this case, the average number of reviews of $C_{50}^{(1)}$ is about 17. Since the size of $S_{50}^{(1)}$ was 156 as mentioned above, about 9 ($\fallingdotseq 156 / 17$) places can be included in $S_{50}^{(1)}$.

$$\nu(S_b^{(i)}) = |S_b^{(i)}| \gamma |C_b^{(i)}| / \gamma(C_b^{(i)}) \tag{4}$$

As summarized, the basic idea to construct $S_b^{(i)}$ is to add a place of $C_b^{(i)}$ whose number of reviews is $\mu(C_b^{(i)})$, until the number of places added to $S_b^{(i)}$ reaches $\nu(S_b^{(i)})$. In the example of the restaurants in Ulsan Namgu, 9 places with 17 reviews can be selected from $C_{50}^{(1)}$, and then be included in $S_{50}^{(1)}$. Other data sets including $S_{100}^{(1)}$ and above can be constructed using the same process.

The number of places with the number of reviews $\mu(C_b^{(i)})$ might be smaller than $\nu(S_b^{(i)})$. In this case, a place whose number of reviews is close to $\mu(C_b^{(i)})$ is added to $S_b^{(i)}$. Thus, the priority for the selection of a place p_j in $C_b^{(i)}$ can be described as follows: the smaller the value of φ_j, the higher the priority of the place p_j.

$$\varphi_j = abs(|R_j| - \mu(C_b^{(i)})) \tag{5}$$

Figure 3 shows an algorithm for constructing $S_b^{(i)}$ from $C_b^{(i)}$. It receives $C_b^{(i)}$ and M as inputs, and generates $S_b^{(i)}$ with a size larger than $\rho(C_b^{(i)}) \times M$. Step 1 initiates $S_b^{(i)}$ to an empty set. Step 2 calculates φ_j for all places p_j in $C_b^{(i)}$. Step3 sorts the places in an increasing order of their φ_j. Steps 4 to 7 add the reviews of place p_j to $S_b^{(i)}$ until the size of $S_b^{(i)}$ is larger than $\rho(C_b^{(i)}) \times M$. After the loop, it returns $S_b^{(i)}$ as a result.

Algorithm 1. Constructing $S_b^{(i)}$ from $C_b^{(i)}$

Input: $C_b^{(i)}$, M
Output: $S_b^{(i)}$
$S_b^{(i)} \leftarrow \emptyset$
Calculate φ_j for all p_j in $C_b^{(i)}$
Sort $C_b^{(i)}$ in an increasing order of φ_j
For each p_j in $C_b^{(i)}$
    Add reviews in R_j to $S_b^{(i)}$
    If $|S_b^{(i)}| > \rho(C_b^{(i)}) \times M$, break
End For
Return $S_b^{(i)}$

FIGURE 3. Algorithm for constructing the experimental data set $S_b^{(i)}$ from the collected source data $C_b^{(i)}$.

		Namgu	Dongu	Bukgu	Ulju	Joonggu	Avg.
Restaurants	Noise	0.554	0.578	0.492	0.557	0.589	**0.554**
	Legitimate	0.446	0.422	0.508	0.443	0.411	**0.446**
Tour-spots	Noise	0.742	0.725	0.733	0.768	0.708	**0.735**
	Legitimate	0.258	0.275	0.267	0.232	0.292	**0.265**

FIGURE 4. Ratio of the noise and legitimate reviews of the restaurants and the tourist attractions in S.

Table 2 shows the number of places included in $S_b^{(i)}$ for each group in Ulsan generated by Algorithm 1. A total of 162 places were included in the data sets. Table 3 shows the number of reviews included in $S_b^{(i)}$ for each group. Total 48,863 reviews were included in the data sets.

Figure 4 shows the ratio of noise and legitimated reviews in S. The average percentage of noise reviews for the restaurants was 55.4%, while the average for the tourist attractions was 73.5%. This shows that the majority of the collected reviews is noise in both of the restaurants and the tourist attractions, and in order to obtain high accuracy, it is essential to detect and filter out the noise reviews before the analysis.

C. MODEL IMPLEMENTATION

In the proposed method, machine learning was adopted to implement a model for the detection of noise reviews. Since reviews are written in text, LSTM, BERT, and DistilBERT were used, which are known to provide high accuracy in text processing.

Together with RNN, LSTM achieves high accuracy in natural language processing by reflecting the order of words in sentences in machine learning. RNN was excluded from the discussion because it has a problem in which the accuracy drops rapidly when the length of sentences increases. BERT is built by pre-training a wide range of text data based on the transformer architecture. In general-purpose natural language processing, BERT is known to provide better performance than LSTM. On the other hand, for the special-purpose applications with limited resources, LSTM can be used more properly, which will be discussed below.

The LSTM model was implemented using *Keras* [56], an open-source software library for the artificial neural networks provided by Google. It connects Embedding, LSTM, and two Dense layers sequentially as follows.

```
model = keras.Sequential([
    keras.layers.Embedding(2000, 50, input_length = 40),
    keras.layers.LSTM(32),
    keras.layers.Dense(128, activation = 'relu'),
    keras.layers.Dense(1, activation = 'sigmoid')
])
```

The Embedding layer contains information about *corpus*, a set of unique words appearing in user reviews. Each word in the corpus is represented as a multi-dimensional vector to maintain its order or relationship with other words. From this, the layer receives three parameters, including the maximum size of the corpus, the dimension of words, and the maximum length of sentences. In the above code, 2000, 50, and 40 were entered for the three parameters, respectively, whose sizes are sufficient to store the experimental data shown in Table 3. In the data, the average number of reviews per place was 695, and each review consisted of 40 words on the average. The average size of corpus per place was 1,768.

Given the three parameters, each word is expressed as a 50-dimensional vector, and each review is delivered in the form of a 40 $\times$ 50 matrix to the LSTM layer. The layer receives each review and performs training to remember the sequence of words in the review. The parameter 32 denotes the number of intermediate nodes used for the training. It can be set to a larger value if more information needs to be trained.

The remaining two Dense layers are used for classification to determine whether a review transferred from the previous layer is noisy or not. They have a general neural network structure. The former has 128 intermediate nodes to store data, i.e., weights and bias, necessary for the classification. Those values are passed to the next layer through the Relu activation function. The latter sums up the passed values and applies the sigmoid function for the classification.

LSTM uses only the data given as input for training. Thus, the size of parameters used for the training is relatively small compared to BERT. For example, the number of the training parameters used in the Embedding layer of the LSTM model can be calculated by multiplying the corpus size and the dimension size of the word matrix, that is, 100,000 parameters (= 2,000 $\times$ 50). On the other hand, BERT uses pre-trained data in addition to a given input data for the training. When the multilingual base model of BERT is used, about 91 million parameters are used for the embedding layer. This means that BERT keeps richer information in the corpus, which becomes the basis for providing higher accuracy in the general natural language applications.

The BERT model was implemented using *ktrain* [57], a light-weight wrapper for the Keras library to help build, train, and deploy neural network models. To use BERT in ktrain, the *text* module is required to be imported. Transformer() function is then used to build a model with the

TABLE 3. Number of reviews included in the experimental data sets $S_b^{(i)}$ for group $G^{(i)}$ in Ulsan generated by Algorithm 1.

	Namgu		Dongu		Bukgu		Ulju		Joonggu	
	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots	Restaurants	Tour-spots
	$G^{(1)}$	$G^{(2)}$	$G^{(3)}$	$G^{(4)}$	$G^{(5)}$	$G^{(6)}$	$G^{(7)}$	$G^{(8)}$	$G^{(9)}$	$G^{(10)}$
$S_{50}^{(i)}$	183	0	151	0	202	0	543	5	158	0
$S_{100}^{(i)}$	186	0	222	0	165	0	194	92	185	0
$S_{250}^{(i)}$	579	0	617	197	453	231	729	200	534	217
$S_{500}^{(i)}$	778	585	400	223	542	264	0	0	854	274
$S_{1000}^{(i)}$	4,165	3,960	4,345	4,182	4,380	4,318	1,169	4,419	4,127	3,835
$S^{(i)}$	5,891	4,545	5,735	4,602	5,742	4,813	2,635	4,716	5,858	4,326

parameter *bert-base-multilingual-uncased* to accept Korean reviews as inputs. The parameter *maxlen* denotes the maximum number of input sentences and was set to 40 which is the same as in the LSTM model.

```
from ktrain import text
t = text.Transformer('bert-base-multilingual-uncased',
                     maxlen= 40, ...)
model = t.get_classifier()
```

While BERT provides better accuracy compared to LSTM, it requires a lot of processing time due to the large number of training parameters. DistilBERT improved its performance by adopting a technique called *knowledge distillation* [21] in the pre-training phase. It is known to be 60% faster than BERT with 40% less memory while achieving 97% accuracy. The DistilBERT model was also implemented using ktrain. The only difference from the above codes lies in the model name used.

```
from ktrain import text
t = text.Transformer('distilbert-base-multilingual-cased',
                     maxlen = 40, ...)
model = t.get_classifier()
```

IV. RESULTS

This section compares the performance of the three models implemented with LSTM, BERT, and DistilBERT through experiments in relation to ③ and ④ discussed in Section I. To measure the accuracy of the models, *f1-score* was used which provides a harmonic average for the skewed data. Experiments were performed on the Google Colab with A100 GPU and 40GB of memory. TensorFlow 2.12.0 was used.

A. INDIVIDUAL LEARNING

The performance of the three models was first checked when the individual learning was performed. For the experiments, data sets shown in Table 2 and 3 were used. For a given place, its reviews were first split into the training and test data sets, whose percentages were 80 and 20, respectively. Using the training data set, learning was performed to obtain a binary classifier to determine whether a given review is noisy. After the training, each review in the test data set was checked using the classifier, and the f1-score was calculated based on the test results.

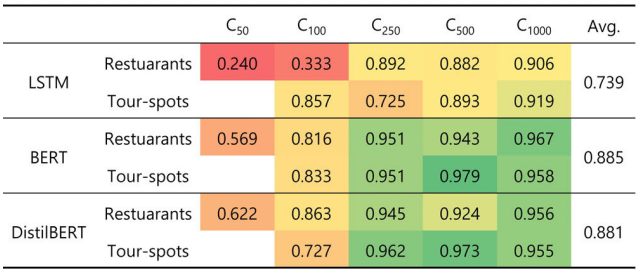

FIGURE 5. Average f1-scores of the LSTM, BERT, and DistilBERT models for restaurants and tourist attractions in Table 2.

Figure 5 compares the average f1-scores of the three models. C_b denotes the set of places in $S_b^{(i)}$ for all $G^{(i)}$s ($1 \le i \le 10$), which is common in all figures below. In the experiment, the BERT model showed the best performance, whose average f1-score was 0.885. The BERT and DistilBERT models showed similar performance. On the other hands, the performance of the LSTM model was poor compared to these two models. Especially for the restaurants in C_{50} and C_{100}, their f1-scores were less than 0.5. These results showed that BERT is more suitable when individual learning is performed.

Figure 6 compares the execution time of the three models when using A100 GPU. The DistilBERT model showed the best performance whose execution time for restaurant and tourist attractions were 22.96 and 24.02 seconds, respectively. On the other hand, the execution time of the BERT model was the largest. The performance gap between the two models was about 36%. The performance of the LSTM model was worse than the DistilBERT model due to the large number of training epochs; 200 epochs were required in the LSTM model to achieve accuracy shown in Figure 5, whereas only 20 epochs were required in other models.

When using a CPU, the execution patterns of the three models change significantly. The execution time of the BERT and DistilBERT models increased drastically, whereas the performance of the LSTM model did not change a lot. Figure 7 compares the execution time of the models when using an Intel-i7 CPU with 16 GB memory. In this case, the

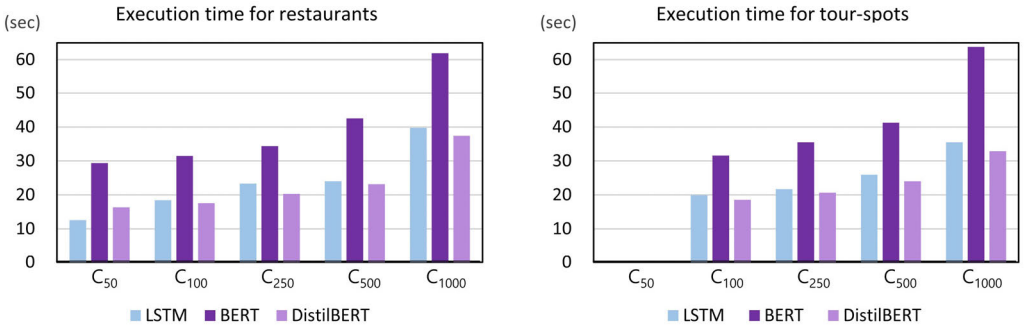

FIGURE 6. Average execution time of the LSTM, BERT, and DistilBERT models for restaurants and tourist attractions in Table 2 when using A100 GPU with 40GB memory.

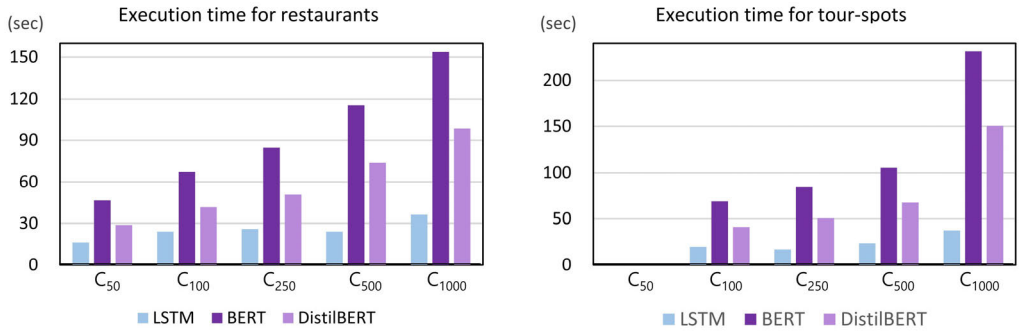

FIGURE 7. Average execution time of the LSTM, BERT, and DistilBERT models for restaurants and tourist attractions in Table 2 when using Intel-i7 CPU with 16 GB memory.

BERT and DistilBERT models required the same number of training epochs to achieve accuracy shown in Figure 5. On the other hand, the number was reduced to 50 in the LSTM model. From this, the execution time of the LSTM model was more than 3 times faster than the BERT model. The gap increased to 5 times for the tourist attractions which have more reviews per place.

We then checked which model is more suitable for the noise review detection for the places with insufficient reviews. For this purpose, f1-score and *Zero-R* values were compared for each place. Zero-R is the simplest classification method which relies on the target; its classifier simply predicts the majority class. For example, if the percentages of noise and legitimated reviews for a place are 65% and 35%, respectively, Zero-R always predicts that a review is noisy, and its accuracy becomes 65%. In this way, Zero-R is useful to determine a baseline performance as a benchmark for other classification methods. The prediction accuracy of the above models also becomes meaningful only when their accuracy is larger than the Zero-R prediction value.

Figure 8 shows the number of places whose f1-scores are less than their Zero-R values in the three models. In the LSTM model, 48 places did not satisfy the Zero-R criteria, which corresponded to 42.1% of the total. Most of the places were included in the class C_{50} and C_{100}. On the other hand, in the BERT and DistilBERT models, 20∼21 places corresponding to about 18% of the total did not meet the criteria. Although the results are superior to the LSTM model, their performance is still not satisfactory.

		C_{50}	C_{100}	C_{250}	C_{500}	C_{1000}	Total
# of restaurants in S_b		55	12	18	8	21	114
Restaurants	LSTM	40	7	0	0	1	48
	BERT	20	1	0	0	0	21
	DistilBERT	20	0	0	0	0	20

FIGURE 8. Number of restaurants in Table 2 whose f1-scores are less than their Zero-R values in the LSTM, BERT, and DistilBERT models.

B. GROUP LEARNING

As shown in Figure 8, for the places with insufficient reviews, it is difficult to expect high accuracy due to the lack of reviews required for training. To resolve the problem, the group learning is proposed where a classifier trained on the reviews of other places with similar characteristics in a group is used for the detection of noise reviews for the places with insufficient reviews in the same group. To conduct the experiments, the places were grouped according to their districts and categories as shown in Table 2.

The data set $S^{(i)}$ constructed from a group $G^{(i)}$ can be used to train and test reviews of the places in $G^{(i)}$. For example, for the places in $G^{(1)}$, the restaurants in Ulsan Namgu, a classifier can be obtained by learning 4,165 reviews of 5 places

belonging to $S^{(1)}_{1000}$. The classifier can then be used for the noise review detection of other places in $S^{(1)}_{50}$ or $S^{(1)}_{100}$. It can also be used for testing reviews of any places in $G^{(1)}$ if their reviews are labeled properly.

As described in the example, we used the reviews of the places belonging to $S^{(i)}_{1000}$ as a training data set for the group $G^{(i)}$. A classifier obtained from the data is then used for the noise review detection of other places in $G^{(i)}$. As a training data, it is possible to use reviews of other places that does not belong to $S^{(i)}_{1000}$. Regarding which data to use for the training, more research is also needed.

Unlike the individual learning, the group learning requires the following two preprocessing steps for the reviews of the places in $G^{(i)}$.

- All place names appearing in the reviews of the places in $G^{(i)}$ should be unified. In the above example, in order for the reviews of the 5 places in $S^{(1)}_{1000}$ to look alike reviews of one place, the names of the 5 places appearing in the reviews need to be changed to one common name. That name must be defined as a special word which does not appear in the corpus of the reviews, such as __NXXX, to avoid conflicts.
- Only words that appear more than T times in the training data sets are extracted and included in the corpus. Each review is then expressed only with the words. This is to increase the similarity of the reviews in a group. As T increases, the similarity of reviews increases, whereas the number of words included in each review decreases, resulting in that the expressiveness of the review is reduced. Therefore, setting T higher does not always lead to the higher accuracy.

To conduct the experiments of the group learning, the above preprocessing steps were performed for the reviews in all data sets $S^{(i)}$ shown in Table 3. In the experiments, T was set to 5, which is an empirical value and may change depending on the application or data characteristics.

Figure 9 compares the average f1-scores of the LSTM model for the places in Table 2 when using the individual and group learning respectively. The group learning showed superior performance compared to the individual learning. Its average f1-score was 0.930, showing 19.1% performance enhancement. Figure 10 and 11 compare the average f1-scores of the BERT and DistilBERT models when using the individual and group learning respectively. In these models, the group learning also showed better performance, and their average f1-scores were similar to that of the LSTM model.

We also checked how much performance improvement was achieved for the places with less than 100 reviews by using the group learning. Figure 12 compares the number of places whose f1-scores are less than their Zero-R values when using the two approaches. In the LSTM model, the number of places was reduced from 48 to 10 when the group learning was applied. In the BERT model, the number was reduced from 21 to 9. In the DistilBERT model, the performance improvement was similar. These results show that the accuracy can be significantly improved by adopting the group learning in the noise review detection.

		C_{50}	C_{100}	C_{250}	C_{500}	C_{1000}	Avg.
Individual learning	Restuarants	0.240	0.333	0.892	0.882	0.906	0.739
	Tour-spots		0.857	0.725	0.893	0.919	
Group learning	Restuarants	0.800	0.941	0.932	0.961	0.970	0.930
	Tour-spots		0.833	0.992	0.972	0.963	

FIGURE 9. Average f1-scores of the LSTM model for places in Table 2 when using the individual and group learning.

		C_{50}	C_{100}	C_{250}	C_{500}	C_{1000}	Avg.
Individual learning	Restuarants	0.569	0.816	0.951	0.943	0.967	0.885
	Tour-spots		0.833	0.951	0.979	0.958	
Group learning	Restuarants	0.818	0.933	0.926	0.982	0.964	0.932
	Tour-spots		0.857	0.966	0.983	0.963	

FIGURE 10. Average f1-scores of the BERT model for places in Table 2 when using the individual and group learning.

		C_{50}	C_{100}	C_{250}	C_{500}	C_{1000}	Avg.
Individual learning	Restuarants	0.622	0.863	0.945	0.924	0.956	0.881
	Tour-spots		0.727	0.962	0.973	0.955	
Group learning	Restuarants	0.817	0.931	0.920	0.980	0.967	0.931
	Tour-spots		0.857	0.976	0.962	0.964	

FIGURE 11. Average f1-scores of the DistilBERT model for places in Table 2 when using the individual and group learning.

		C_{50}	C_{100}	C_{250}	C_{500}	C_{1000}	Total
# of restaurants in S_b		55	12	18	8	21	114
Individual learning	LSTM	40	7	0	0	1	48
	BERT	20	1	0	0	0	21
	DistilBERT	20	0	0	0	0	20
Group learning	LSTM	9	1	0	0	0	10
	BERT	8	0	1	0	0	9
	DistilBERT	8	0	1	0	0	9

FIGURE 12. Number of restaurants in Table 2 whose f1-scores are less than their Zero-R values when using the individual and group learning in the LSTM, BERT, DistilBERT models.

C. DISCUSSION

Figure 13 summarizes the experimental results in terms of the overall performance of the individual and group learning when using the LSTM, BERT, and DistilBERT models over the data set S. The left figure compares the average f1-scores of the models obtained from all places in S. The middle shows the average scores from the places with less than 100 reviews. The right shows the average scores from the places with larger than or equal to 100 reviews. From the figures, the following results were observed.

	LSTM	BERT	DistilBERT	Avg.
Individual learning	0.739	0.885	0.881	0.835
Group learning	0.930	0.932	0.931	0.931

	LSTM	BERT	DistilBERT	Avg.
Individual learning	0.477	0.740	0.737	0.651
Group learning	0.858	0.869	0.868	0.865

	LSTM	BERT	DistilBERT	Avg.
Individual learning	0.870	0.958	0.953	0.927
Group learning	0.965	0.964	0.962	0.964

FIGURE 13. Average f1-scores of the LSTM, BERT, and DistilBERT models when using the individual and group learning: (a) for all places (left), (b) for places with less than 100 reviews (middle), and (c) for places with larger than or equal to 100 reviews (right).

- The group learning provided better performance than the individual learning. The average f1-score of the group learning was 0.931, which is superior than the result of the individual learning, 0.835.
- In the group learning, all three models showed similar performance, where the gap between the models was within 1%.
- For the places with less than 100 reviews, the detection accuracy was improved dramatically by using the group learning, which was from 0.651 to 0.865.
- For the places with sufficient reviews, the group learning showed very high accuracy. The average f1-score of the three models was 0.964.

Note that in the above results, the performance of the group learning was similar in all three models. This shows that, if the training data is sufficient as in the group learning, LSTM can provide good performance. In addition, the execution time of the LSTM model was considerably smaller than those of the other models based on BERT, as shown in Figure 6 and 7. As a result, the LSTM model can be a reasonable solution for the proposed group learning. If computation resources are limited, e.g., when GPU is not available, it becomes more attractive. When the training data increases and it is required to distribute the data into multiple nodes, the resource allocation techniques discussed in [58] and [59] can be employed.

V. CONCLUSION

This paper proposed the group learning to perform the noise review detection efficiently over a large number of places and their reviews. In the method, multiple places can be grouped according to their similarity, e.g., located in the same region or belonging to the same business category. A classifier trained on the reviews of the places belonging to a group can be used for the noise review detection of other places with insufficient reviews in the group. As an initial study of this approach, we tried to check through experiments whether the proposed group learning can provide higher accuracy than the conventional individual learning over the real data. For the experiments, about 7,651 places and 1.6 million reviews were collected online. To perform supervised learning, the training and test data sets were sampled from the data, which can reflect the characteristics of the collected data properly. To build the data sets, the criteria related to ① and ② discussed in Section I were determined heuristically as follows.

① As a criterion for grouping places, regions and categories were used. From this, the places were divided into 10 groups, including the restaurant and tourist attraction groups for each district in Ulsan.

② For each group, about 5,000 reviews were sampled to conduct supervised learning, whose size is small enough to be manually labeled.

From the above, about 50,000 reviews of 162 places were included in the experimental data sets. To implement a model for the detection of noise reviews, LSTM, BERT, and Distil BERT were employed, which are known to provide high accuracy in text processing. Regarding ③ and ④ discussed in Section I, the following results were observed through experiments using the data sets and models.

③ As shown in Figure 13 (a), the performance of the group learning was superior than that of the individual learning. In particular, for the places with sufficient reviews, the group learning showed very high accuracy as shown in Figure 13 (c), whose average f1-score was 0.964.

④ As shown in Figure 13 (b), for the places with less than 100 reviews, the accuracy was significantly improved by using the group learning, which was from 0.651 to 0.865. This shows that the group learning can be used properly for the places where the learning is difficult due to the lack of reviews.

The above results show that the group learning proposed in this paper can be applied properly to the problem of the noise review detection when a large number of places and reviews need to be analyzed. One thing to note in the results is that the performance of the group learning was similar in all three models. This indicates that LSTM can be a reasonable solution for the proposed group learning. If resources are limited, it becomes more attractive.

In future, regarding ①, we plan to study the relationship between the grouping criteria and detection accuracy more in depth. Regarding ②, a data sampling method to maximize the performance of the group learning will be investigated. Also, the research on how to increase accuracy for the places with insufficient reviews will be continued.

REFERENCES

[1] N. Sachdeva and J. McAuley, "How useful are reviews for recommendation? A critical review and potential improvements," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1845–1848.

[2] Z. Xu, H. Zeng, and Q. Ai, "Understanding the effectiveness of reviews in e-commerce top-N recommendation," in *Proc. ACM SIGIR Int. Conf. Theory Inf. Retr.*, Jul. 2021, pp. 149–155.

[3] J. Bao, Y. Zheng, D. Wilkie, and M. Mokbel, "Recommendations in location-based social networks: A survey," *GeoInformatica*, vol. 19, no. 3, pp. 525–565, Jul. 2015.

[4] I. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso, "Where shall we go today: Planning touristic tours with tripbuilder," in *Proc. 22nd ACM Int. Conf. Conf. Inf. Knowl. Manage. (CIKM)*, 2013, pp. 757–762.

[5] Y.-T. Wen, J. Yeo, W.-C. Peng, and S.-W. Hwang, "Efficient keyword-aware representative travel route recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 8, pp. 1639–1652, Aug. 2017.

[6] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst.*, Nov. 2012, pp. 199–208.

[7] H. G. Kim, "Developing a big data analysis platform for small and medium-sized enterprises," *J. KSCI*, vol. 25, no. 8, pp. 65–72, 2020.

[8] N. Jindal and B. Liu, "Analyzing and detecting review spam," in *Proc. 7th IEEE Int. Conf. Data Mining (ICDM)*, Oct. 2007, pp. 547–552.

[9] N. Jindal and B. Liu, "Opinion spam and analysis," in *Proc. Int. Conf. Web Search Web Data Mining (WSDM)*, 2008, pp. 219–230.

[10] M. Crawford, T. M. Khoshgoftaar, J. D. Prusa, A. N. Richter, and H. Al Najada, "Survey of review spam detection using machine learning techniques," *J. Big Data*, vol. 2, no. 1, pp. 1–24, Dec. 2015.

[11] Y. Ren and D. Ji, "Learning to detect deceptive opinion spam: A survey," *IEEE Access*, vol. 7, pp. 42934–42945, 2019.

[12] L. He, X. Wang, H. Chen, and G. Xu, "Online spam review detection: A survey of literature," *Hum.-Centric Intell. Syst.*, vol. 2, nos. 1–2, pp. 14–30, Jun. 2022.

[13] A. Mewada and R. K. Dewang, "A comprehensive survey of various methods in opinion spam detection," *Multimedia Tools Appl.*, vol. 82, no. 9, pp. 13199–13239, Apr. 2023.

[14] R. Y. K. Lau, S. Y. Liao, R. C.-W. Kwok, K. Xu, Y. Xia, and Y. Li, "Text mining and probabilistic language modeling for online review spam detection," *ACM Trans. Manage. Inf. Syst.*, vol. 2, no. 4, pp. 1–30, Dec. 2011.

[15] J. A. Diaz-Garcia, M. D. Ruiz, and M. J. Martin-Bautista, "NOFACE: A new framework for irrelevant content filtering in social media according to credibility and expertise," *Expert Syst. Appl.*, vol. 208, Dec. 2022, Art. no. 118063.

[16] C. Pezoa-Fuentes, D. García-Rivera, and S. Matamoros-Rojas, "Sentiment and emotion on Twitter: The case of the global consumer electronics industry," *J. Theor. Appl. Electron. Commerce Res.*, vol. 18, no. 2, pp. 765–776, Mar. 2023.

[17] R. Patel and K. Passi, "Sentiment analysis on Twitter data of world cup soccer tournament using machine learning," *IoT*, vol. 1, no. 2, pp. 218–239, Oct. 2020.

[18] *Naver Blog Search API*. Accessed: Oct. 16, 2023. [Online]. Available: https://developers.naver.com/docs/serviceapi/search/blog/blog.md

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[20] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. (2018), "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[21] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," 2019, *arXiv:1910.01108*.

[22] E. F. Cardoso, R. M. Silva, and T. A. Almeida, "Towards automatic filtering of fake reviews," *Neurocomputing*, vol. 309, pp. 106–116, Oct. 2018.

[23] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, "Finding deceptive opinion spam by any stretch of the imagination," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics*, 2011, pp. 309–319.

[24] M. Ott, C. Cardie, and J. T. Hancock, "Negative deceptive opinion spam," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2013, pp. 497–501.

[25] S. Shojaee, M. A. A. Murad, A. B. Azman, N. M. Sharef, and S. Nadali, "Detecting deceptive reviews using lexical and syntactic features," in *Proc. 13th Int. Conf. Intellient Syst. Design Appl.*, Dec. 2013, pp. 53–58.

[26] A. Abbasi, H. Chen, and J. F. Nunamaker, "Stylometric identification in electronic markets: Scalability and robustness," *J. Manage. Inf. Syst.*, vol. 25, no. 1, pp. 49–78, Jul. 2008.

[27] S. Feng, R. Banerjee, and Y. Choi, "Syntactic stylometry for deception detection," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2012, pp. 171–175.

[28] J. Li, M. Ott, C. Cardie, and E. Hovy, "Towards a general rule for identifying deceptive opinion spam," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 1566–1576.

[29] J. Eisenstein, A. Ahmed, and E. P. Xing, "Sparse additive generative models of text," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1041–1048.

[30] T. Qian and B. Liu, "Identifying multiple userids of the same author," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 1124–1135.

[31] F. Li, M. Huang, Y. Yang, and X. Zhu, "Learning to identify review spam," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 2488–2493.

[32] A. Mukherjee, B. Liu, and N. Glance, "Spotting fake reviewer groups in consumer reviews," in *Proc. 21st Int. Conf. World Wide Web*, Apr. 2012, pp. 191–200.

[33] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance, "What yelp fake review filter might be doing?" in *Proc. 7th Int. AAAI Conf. Weblogs Social Media*, vol. 2013, pp. 409–418.

[34] H. Li, Z. Chen, A. Mukherjee, B. Liu, and J. Shao, "Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns," in *Proc. 9th Int. AAAI Conf. Weblogs Social Media*, 2015, pp. 634–637.

[35] A. Barushka and P. Hajek, "Review spam detection using word embeddings and deep neural networks," in *Proc. AIAI*, 2019, pp. 340–350.

[36] L. Li, W. Ren, B. Qin, and T. Liu, "Learning document representation for deceptive opinion spam detection," in *Proc. 14th China Nat. Conf. Comput. Linguistics*, 2015, pp. 393–404.

[37] K. O'Shea and R. Nash, "An introduction to convolutional neural network," 2015, *arXiv:1511.08458*.

[38] S. Zhao, Z. Xu, L. Liu, M. Guo, and J. Yun, "Towards accurate deceptive opinions detection based on word order-preserving CNN," *Math. Problems Eng.*, vol. 2018, pp. 1–9, Sep. 2018.

[39] G. M. Shahariar, S. Biswas, F. Omar, F. M. Shah, and S. Binte Hassan, "Spam review detection using deep learning," in *Proc. IEEE 10th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Oct. 2019, pp. 0027–0033.

[40] C.-C. Wang, M.-Y. Day, C.-C. Chen, and J.-W. Liou, "Detecting spamming reviews using long short-term memory recurrent neural network framework," in *Proc. 2nd Int. Conf. E-Commerce, E-Bus. E-Government*, Jun. 2018, pp. 16–20.

[41] W. Liu, W. Jing, and Y. Li, "Incorporating feature representation into BiLSTM for deceptive review detection," *Computing*, vol. 102, no. 3, pp. 701–715, Mar. 2020.

[42] X. Wang, K. Liu, and J. Zhao, "Detecting deceptive review spam via attention-based neural networks," in *Proc. 6th Conf. Natural Lang. Process. Chin. Comput.*, 2017, pp. 866–876.

[43] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[44] Y. Ren and Y. Zhang, "Deceptive opinion spam detection using neural network," in *Proc. 26th Int. Conf. Comput. Linguistics*, 2016, pp. 140–150.

[45] L. R. Medsker and L. C. Jain, "Recurrent neural networks," *Des. Appl.*, vol. 5, pp. 64–67, Dec. 2001.

[46] P. Bhuvaneshwari, A. N. Rao, and Y. H. Robinson, "Spam review detection using self attention based CNN and bi-directional LSTM," *Multimedia Tools Appl.*, vol. 80, no. 12, pp. 18107–18124, May 2021.

[47] G. Bathla, P. Singh, R. K. Singh, E. Cambria, and R. Tiwari, "Intelligent fake reviews detection based on aspect extraction and analysis using deep learning," *Neural Comput. Appl.*, vol. 34, no. 22, pp. 20213–20229, Nov. 2022.

[48] S. Kanmani and S. Balasubramanian, "Leveraging readability and sentiment in spam review filtering using transformer models," *Comput. Syst. Sci. Eng.*, vol. 45, no. 2, pp. 1439–1454, 2023.

[49] Y. Liu, M. Ott, N. Goyal, J. Du, and M. Joshietal, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[50] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. NeurIPS*, 2019.

[51] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, and G. Wenzek, "Unsupervised cross-lingual representational learning at scale," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 8440–8451.

[52] R. A. Duma, Z. Niu, A. S. Nyamawe, J. Tchaye-Kondi, and A. A. Yusuf, "A deep hybrid model for fake review detection by jointly leveraging review text, overall ratings, and aspect ratings," *Soft Comput.*, vol. 27, no. 10, pp. 6281–6296, May 2023.

[53] *Small Business Promotion Agency*. Accessed: Oct. 16, 2023. [Online]. Available: https://www.semas.or.kr

[54] *Tour API 3.0, Korea Tourism Organization*. Accessed: Oct. 16, 2023. [Online]. Available: https://kto.visitkorea.or.kr/kor/gov30/tourapi.kto
[55] *Naver Local Search API*. Accessed: Oct. 16, 2023. [Online]. Available: https://developers.naver.com/docs/serviceapi/ search/local/local.md
[56] *Google Keras*. Accessed: Oct. 16, 2023. [Online]. Available: https://keras.io
[57] A. S. Maiya, "ktrain: A low-code library for augmented machine learning," *J. Mach. Learn. Res.*, vol. 23, no. 1, pp. 7070–7075, 2022.
[58] H. Ahmadvand, T. Dargahi, F. Foroutan, P. Okorie, and F. Esposito, "Big data processing at the edge with data skew aware resource allocation," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2021, pp. 81–86.
[59] H. Ahmadvand, F. Foroutan, and M. Fathy, "DV-DVFS: Merging data variety and DVFS technique to manage the energy consumption of big data processing," *J. Big Data*, vol. 8, no. 1, pp. 1–16, Dec. 2021.

HYEON GYU KIM received the B.S. and M.S. degrees in computer science from the University of Ulsan, in 2000, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2010.

From 2001 to 2011, he was a Chief Research Engineer with LG Electronics. Since 2012, he has been an Associate Professor with the Division of Computer Science and Engineering, Sahmyook University, Seoul, South Korea. His research interests include artificial intelligence and big data processing.

YOO HYUN PARK received the B.S., M.S., and Ph.D. degrees in computer science from Pusan National University, Busan, South Korea, in 1996, 1998, and 2008, respectively. From 2001 to 2009, he was a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. Since 2009, he has been a Professor with the Division of Computer Software Engineering, Dong-Eui University, Busan. His research interests include cloud computing, big data analysis, the IoT healthcare, and computer education.