IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 13, 2020, accepted July 27, 2020, date of publication August 6, 2020, date of current version August 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3014779

Deep Learning-Based Context-Sensitive Spelling Typing Error Correction

JUNG-HUN LEE[1], MINHO KIM[2], AND HYUK-CHUL KWON[3]
[1]Department of Electrical and Computer Engineering, Pusan National University, Busan 46241, South Korea
[2]Department of Software, Catholic University of Pusan, Busan 46252, South Korea
[3]Department of Information Computer Science, Pusan National University, Busan 46241, South Korea

Corresponding author: Hyuk-Chul Kwon (hckwon@pusan.ac.kr)

This work was supported by the Institute for Information & communications Technology Planning & Evaluation (IITP) Grant funded by the Korean Government, Ministry of Science and ICT (MSIT) (Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services) under Grant 2013-0-00131.

ABSTRACT This study aims to solve the context-sensitive spelling error problem for English documents. There are two types of spelling errors in English: non-word spelling errors and context-sensitive spelling errors. Non-word spelling errors are simple to correct because they can only be detected by matching the words in sentences with those in a dictionary; however, context-sensitive spelling errors entail increased difficulty of correction because the relationship between the word to be corrected and the surrounding context must be known. Spelling errors are considered noise in every field that uses text information, and preprocessing via document correction is necessary to minimize this problem. Context-sensitive spelling errors include homophone errors (which arise from the incorrect use of words that sound the same but are spelled differently), typographical errors (caused by striking an incorrect key on a keyboard), grammatical errors (which occur when the user does not know the correct grammatical rules), and cross word boundary errors (which arise from incorrect spacing between words). This study focuses on typographical errors. The context-sensitive spelling error problem is solved using the deep learning method, which is not an existing statistical method. The deep learning language model-based correction approach is divided into four parts, namely, correction based on word embedding information, contextual embedding information, an auto-regressive (AR) language model, and an auto-encoding (AE) language model. In this study, the best correction performance was obtained for the AE language model-based approach, and we verified its performance through a detailed correction test.

INDEX TERMS Context-sensitive spelling error correction, natural language processing, word embedding, contextual embedding, auto-regressive, auto-encoding, permutation language model.

I. INTRODUCTION

Spelling errors can be classified into two categories: non-word and context-sensitive spelling errors. The former occur when a word is spelt with a non-conventional spelling, such as ''fron.'' Thus, it is easy to detect these errors by analyzing a word morphologically. An example of the latter error type is when a word such as ''fake'' is used with ''pretty'' to yield ''a pretty fake.'' It is only possible to detect such errors by considering the morphological and semantic characteristics of the words. In Table 1, the four categories of context- sensitive spelling errors are listed: homophone errors, typographical errors, grammatical errors, and cross word boundary errors. In this study, we address typographical errors, which are errors caused by the user incorrectly typing on the keyboard. The details of this error type are described in the subsections related to context-sensitive spelling error correction in Section 3. Notably, it was previously found that context-sensitive spelling errors accounted for 30–40% [1], [2] of the total spelling errors in pre-corrected documents in English. Furthermore, correcting these errors had a significant influence on the overall performance of the spell checker.

The methods used to correct context-sensitive spelling errors can be separated into three categories: rule-based, statistical, and deep learning-based method. Rule-based methods have a high probability of correcting a spelling error with

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

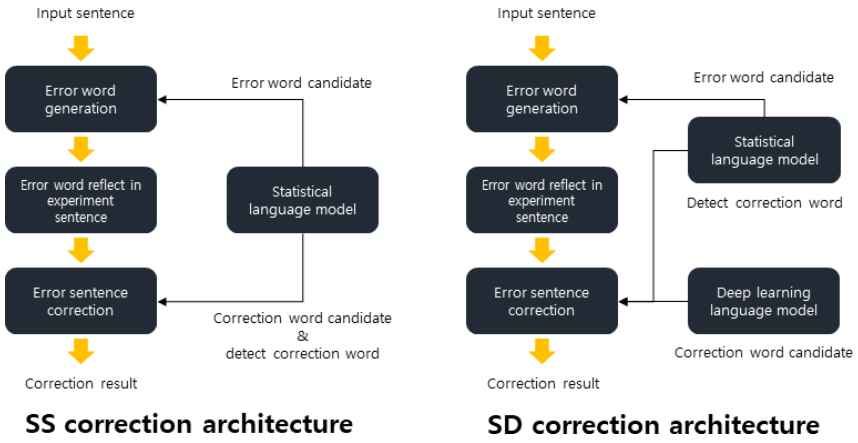


FIGURE 1. Comparison of architecture between a) SS correction and b) SD correction.

TABLE 1. List of context-sensitive spelling error types.

Error type	*Cause of error*	*Example*
homophone error	words that sound that same, but are spelled differently	peace / piece
typographical error	striking an incorrect key on a keyboard	from / form
grammatical error	the user did not know exactly what the difference between grammars	among / between
cross word boundary error	wrong blank between words	maybe / may be

a high incidence or standardized context; however, it is difficult for such methods to correct unstructured errors caused by input errors. In contrast, the statistical method can be applied to context-sensitive spelling errors that have low repeatability, and it is frequently utilized because it can be used to develop spelling error correction technologies suitable for various environments in a short time by changing the corpus used for statistics. Deep learning-based correction can be applied with regard to not only morphological approaches, such as rule-based and statistical methods, but also to gain a deep semantic understanding of context. Figure 1 presents an example of a correction experiment that is conducted in three stages. First, the error-free document to be employed in the correction experiment is input, and a sentence in the document is used to generate the error word, which is replaced by the correct word. The error documents are then created by including the generated error words in the sentences, instead of the correct answer words. Finally, the error document is actually corrected and the performance is measured. At each stage, the language model performs a variety of actions. In the generation stage of the experimental document, the error word is created. In the correction stage, the search for the error word, the generation of the correction candidate word, and the final correction are performed. The block diagram in Figure 1(a) corresponds to the existing statistical language model, which performs all the aforementioned actions. Figure 1(b) presents a method of combining statistics and deep learning. The statistical language model performs the tasks of error word generation and error word search, and the deep learning language model performs the final correction of the error word. As shown in Figure 1, both the statistical method-based error detection model (SS) and deep learning-based correction model (SD) search for error words using a statistical language model because it is difficult to search for errors in documents using a deep learning language model. For this reason, the statistical language model only produces and compares candidate words for co-occurrence words of error search target words. In contrast, all the words learned by the deep learning language model are considered when searching for the target error words. If the deep learning language model is used to search for the error word, the correction speed is slower and the correction accuracy is lower because the entire word of the correction document is judged as an error word and the correction is executed.

In this study, we apply various recently developed deep learning language models to context-sensitive spelling error correction and suggest the direction of a correction experiment. The correction experiment was conducted on typographical errors, and the performance of the model was measured by subdividing the errors that may arise from typing on a keyboard.

This paper is structured as follows: Section 2 presents related research, Section 3 discusses the context-sensitive spelling errors considered in this study, Section 4 elucidates the correctional language model, Section 5 presents an analysis of the experiment and results, and finally, Section 6 presents the conclusion and future research.

II. RELATED RESEARCH

Research on context-sensitive spelling error correction is conducted via two broad methods. The first one entails the method of generating the correction candidate word, and the second employs the relationship between the candidate word and the context to determine the final correction word. The

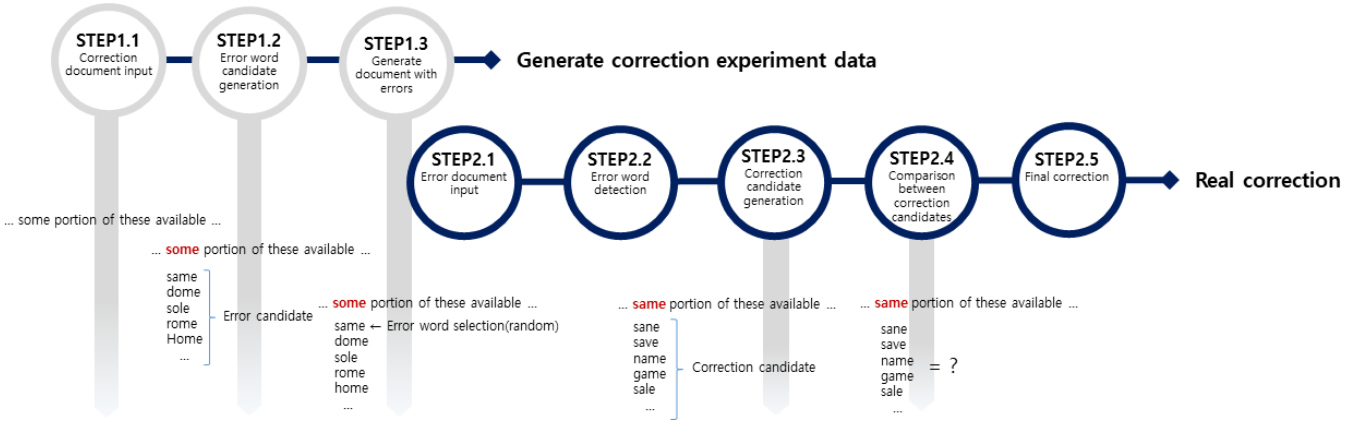


FIGURE 2. Context-sensitive spelling error correction process.

initial candidate word generation method calculates the edit distance between the target correction word and the corresponding dictionary word [3]. Subsequently, it was developed and applied to the method [4] that restricts candidate generation by considering the edit distance and the distance from the corresponding alphabet on the keyboard, based on the keyboard input environment. Recently, a method that employs contextual information [5] was developed to overcome the necessity of comparing words. This method was used in the present study for generating and searching error words. The method of generating candidate words using contextual information is called 3-gram. In the present study, various high-quality candidate words were generated using information extracted from ten quadrillion words contained in the English corpus.

The next research aim is to find the optimal correction candidate, i.e., to develop or select an appropriate correction language model. Correction language models used in context-sensitive spelling error correction have been developed using both the statistical correction method and the deep learning-based correction method. The statistical correction method has been typically employed thus far, such as in the noisy channel model [3] or the n-gram-based language model [6]. The statistical methods that have been researched in the context of the Korean language include smoothing, interpolation, and improvement of the n-gram search structure [4], [5], [7] based on the noisy channel model.

Recently, a correction method has been developed using deep learning, and studies have been conducted on correction models based on recurrent neural networks and convolutional neural networks [8], [9] in addition to corrections performed using word embedding [10], [11]. In recent years, there has been a lack of research on context-sensitive spelling errors; however, the correction of documents is worthy of research because it is substantially advantageous in text-related research or document writing.

The context-sensitive spelling error correction addressed in this paper is conducted on various words, in a wide range of documents. In the context-sensitive spelling error correction process, it is difficult to obtain correct answers to spelling errors for all words; therefore, we chose a deep learning language model based on unsupervised learning. We propose a context-sensitive spelling error correction method using various deep learning language models.

III. CORRECTION OF CONTEXT-SENSITIVE SPELLING ERRORS

A. CONTEXT-SENSITIVE SPELLING CORRECTION PROCEDURE

In this study, context-sensitive spelling error correction is performed in the same order as that shown in Figure 2. The figures shows the steps (1.1–1.3) of creating the error document used in the experiment and the steps (2.1–2.5) of correcting the generated error document or correcting the actual document. Note that when the actual correction is performed, the first three steps are excluded. If included, it is for the correction performance experiment through the correct answer. First, in step 1.1, an accurate document or sentence is input to generate an error sentence to be used for the experiment. In step 1.2, the error word is created for the input sentence. This word is different from the target word, and the extent of this difference is defined as the edit distance, which the experimenter establishes. In the example, the target word is "some" in the sentence "…some portion of these available …," and a set of error words (same, dome, sole, rome, home, etc.) with an edit distance of 1 are produced. In step 1.3, the final error word is randomly selected from the candidate error words. It is important to note that candidate error word generation uses a statistical language model. Steps 1.1–1.3 are repeated, and error words are created for the entire input sentence. When several error words are generated for one sentence, in each experiment, the target word in the answer sentence is replaced according to the selected candidate error word, and the correction experiment is performed independently. Next, in step 2.1, the error sentence

generated in steps 1.1–1.3 is inputted. In steps 2.2 and 2.3, the entire phrase is searched sequentially, without the error word being known. Furthermore, if an appropriate correction word candidate exists within the target edit distance, the correction is performed. In the example, the candidate correction words (sane, save, name, game, sale, etc.) appeared for the sentence "…same portion of these available …" with regard to the error in "same," which is judged as the target correction word. We search for error words using statistical language models, such as error word generation models. In step 2.4, the final correction word is selected by calculating and comparing the distance between the surrounding sentence and the candidate word. In the correction stage, the entire sentence is circulated, and the process of steps 2.2 and 2.3 is repeated. The details regarding the candidate error word generation performed in step 1.2 are explained in Section 5A, and the description of the generation of correction candidate words and the selection of the final correction word in steps 2.2 and 2.3 is provided in Section 3B. Section 4 explains the types of language models used for context-sensitive spelling error correction.

B. CONTEXT-SENSITIVE SPELLING CORRECTION TECHNIQUE

Context-sensitive spelling error correction techniques that employ language models are divided into four main categories: word embedding information based, contextual embedding information based, auto-regressive (AR) language model based, and auto-encoding (AE) language model based correction techniques. In the case of the permutation language model, training is performed via the AR method, but the sentence is shuffled. The bidirectional context information is obtained, and the correction method is applied in the same way as that in the AE method.

First, we formulate context-sensitive spelling error correction based on word embedding information using Equations (1) and (2). In Equation (1), x_i represents a candidate correction word for the target correction word, and the *dis* function sums the inner product values of the words and the window size contexts. The sum of the inner product values is equated with the probability value of the context. The smaller the distance between the target word and the correction word, the higher the probability value. Thus, Equation (1) is a method to obtain the probability of the correction candidate. The symbol i represents the order of the correction candidate words generated for the target correction words selected in the sentence, and C is the maximum number of candidate correction words.

$$argmax\ dis\,(sentence, x_i)\,, \quad (0 \le i \le C) \tag{1}$$

Equation (2) entails the sum of the inner product values consistent with the correction candidate word x_i and the surrounding context in Equation (1). Furthermore, t of x_t is the position of the word in which the whole candidate set C of x_i is located. The term *win* indicates the size of the surrounding context of the correction candidate word, and $cos(x_t, x_j)$ is a function to obtain the inner product between the correction candidate word and the context word. This function applies the distance value between words in the embedding language model. Finally, λ is the smoothing value for out of vocabulary (OOV) cases.

$$dis\,(sentence, x_i) = \sum_{\substack{j=t-win \\ j\neq t}}^{t+win} cos\left(x_t, x_j\right) + \lambda, (x_t = x_i) \tag{2}$$

Equation (2) is typically used in word-by-word embedding (Word2Vec [12], GloVe [13], fastText [14], etc.) in a way that limits the distance of the reference context because a larger distance between the target correction word and the context decreases the correlation. As an example, in the fastText model, which overcomes OOV cases using sub-word information, zero is processed.

Equation (3) is used for the whole sentence, without restriction, for correction based on contextual embedding. With regard to Equation (3), $dis(sentence, x_i)$ computes the inner product for the word x_j with the whole context T, except for the target correction word x_t, and sums the resulting values. Unlike Equation (2), there is no smoothing value λ because context-based embedding uses subwords in the learning process; therefore, OOV processing is performed.

$$dis\,(sentence, x_i) = \sum_{\substack{j=0 \\ j\neq t}}^{T} cos\left(x_t, x_j\right), (x_t = x_i) \tag{3}$$

The following is the correction method based on the AR language model. This is the characteristic of the AR language model, owing to the unidirectional (forward and backward) information used for the correction. The sentence $(x_1, x_2, \cdots, x_T)$ is inputted into the model based on Equation (4). The set of correction candidate words selected in the error search process is called C, and $\hat{C}$ selects $dis(x_1, x_2, \cdots, C, \cdots, x_{T-1}, x_T)$ with the maximum context-sensitive spelling error correction distance value.

$$\hat{C} = \underset{c}{argmax}\ dis\,(x_1, x_2, \cdots, C, \cdots, x_{T-1}, x_T) \tag{4}$$

The set C, which represents the set of candidate correction words with regard to Equation (4), uses the edit distance function (EDF), as in Equation (5), and obtains the candidate words for correction N that satisfy the entire word embedding vocabulary of the mask language model and the set edit distance, based on the central word $x_t(\in C)$. The term V represents the number of whole word embedding vocabularies learned by the language model.

$$\begin{aligned} C &= EDF\left(x_t, embedding\ voca_{1:V}\right) \\ &= candidate_{1:N}\ (V \ge N) \end{aligned} \tag{5}$$

The distance between each candidate and context is obtained using the bi-direction function $BiDF$, given by Equation (6). In $BiDF$, the vector information of the corresponding candidate and the surrounding context, obtained

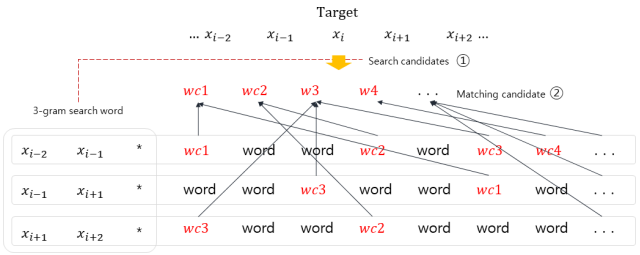


FIGURE 3. Candidate generation method in embedding based correction.

from the forward and backward directions, is summed and compared by considering the unidirectional characteristic, which is the characteristic of the AR language model.

$$\begin{aligned} &dis\,(x_1, x_2, \cdots, C, \cdots, x_{T-1}, x_T) \\ &\quad = BiDF\,(sentence, candidate_{1:N}) \\ &\quad = sum\begin{pmatrix} forward\,(sentence, candidate_{1:N}) \\ , bacward\,(sentence, candidate_{1:N}) \end{pmatrix} \\ &\quad = candidate\ distance_{1:N} \end{aligned} \tag{6}$$

Finally, the correction word is determined by masking the target correction word in the correction sentence using the correction method based on the AE language model. Referring to Equation (4), for a sentence $(x_1, x_2, \cdots, x_T)$ input into a model such as the AR language model, the set of correction candidates selected during the error search process is called C. The set C is obtained via the same method as that elucidated with regard to Equation (5), described in the correction of the AR language model method. Furthermore, $\hat{C}$ selects the $dis(x_1, x_2, \cdots, C, \cdots, x_{T-1}, x_T)$ for which the context-sensitive spelling error correction distance value is maximum. The *dis* function obtains the distance value between each candidate and context using the masked language model function *MLMF* as the masking sentence and N correction candidate words as the input, as in Equation (7).

$$\begin{aligned} &dis\,(x_1, x_2, \cdots, C, \cdots, x_{T-1}, x_T) \\ &\quad = MLMF\,(sentence, candidate_{1:N}) \\ &\quad = candidate\ distance_{1:N} \end{aligned} \tag{7}$$

1) GENERATION OF CANDIDATE WORDS IN CONTEXT-SENSITIVE SPELLING CORRECTION

The candidate correction words should be created based on the words identified as error words using statistical methods. The error search method uses the 3-gram information extracted from the corpus (Google Web 1T [15]) and determines the context based on the location of the correction object word, as shown in Figure 3. The location of the error word search target word is called i, and the element word of the context that can be combined with the 3-gram information has $x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}$. All the words that can potentially appear in the position of x_i will be retrieved from three directions using 3-gram. If the edit distance is not limited, tens of thousands of words may be searched. Therefore, the

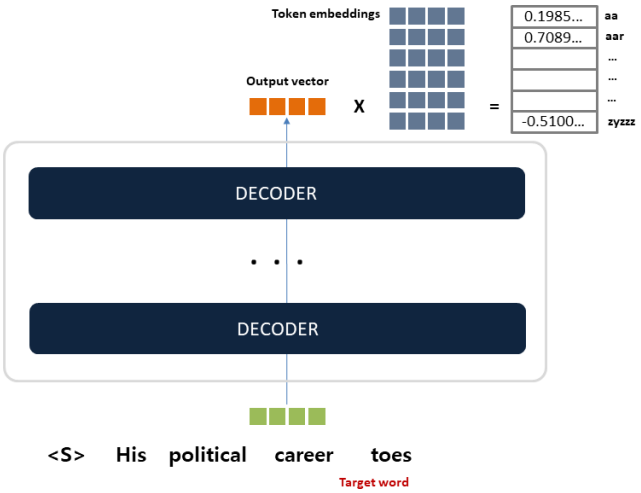


FIGURE 4. Calculation of correction values in GPT-2 [16].

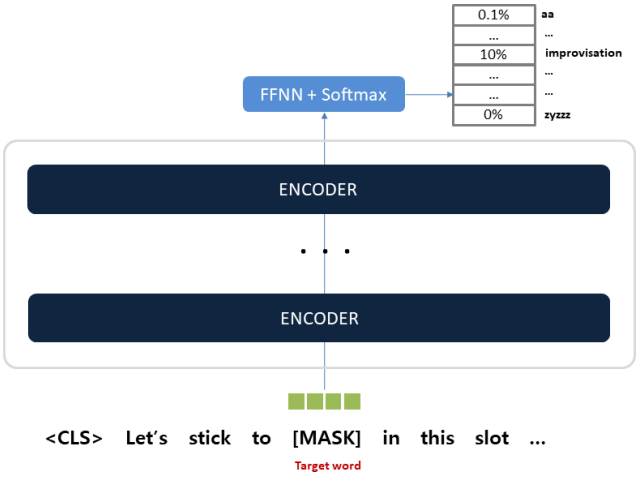


FIGURE 5. Calculation of correction values in BERT [17].

correction candidate is selected by calculating the edit distance with x_i, and the correction is performed if the candidate word corresponding to the target word of the error search exists. This method is the same as that used to generate error words in the documents used in the correction experiment, described in Section 5A, and the methodology is detailed in that section.

In the correction method based on embedding, correction is performed using statistically obtained candidate words. In the other deep learning language model, the candidate words are determined and corrected based on each learning embedding vocabulary. For example, when the target correction word is "toes," "His political career" is entered into the model and the probability value is calculated for the whole embedding vocabulary to predict the word that will appear after "career." For sentence generation, the word with a ranking of 1 should be finally selected. However, for correction, the word with the highest probability value is determined as the final correction word from the words, determined by calculating the target correction word and the edit distance, that are below the preset edit distance. Figure 4 presents an example using Generative

Pre-Training 2 (GPT-2) [16], which is a method that considers only the forward direction. In an AR language model such as GPT-2, the reverse string can be input in the same way as the forward string to refer to the information corresponding to the backward direction; the results can be obtained after the learning process. Figure 5 is a typical example of bidirectional encoder representations from transformers (BERT) [17] as an AE language model, and the probability value is obtained by performing a Softmax calculation for the entire embedding vocabulary by masking the target correction word. The correction word is determined by calculating the edit distance, with the original word masked in the sentence, instead of selecting the word with the highest probability as the correction word, as with GPT-2. To correct a larger volume of information, we can correct the information corresponding to the backward direction by combining the information entailed in Equation (6). Additionally, we can utilize the AE language model using bidirectional information, such as that entailed in Equation (7).

TABLE 2. Classification of typographical errors.

Code	*Classification*	*Example*	*Edit distance*
OX	Omission of a letter	major / maor	1
XO	Addition of a letter	this / thjis	1
DOUB12	Single letter instead of double letter	operation / opperation	1
DOUB21	Double letter instead of single letter	succeeded / succeded	1
XY	Substitution of one letter	continue / continur	1
SWAP	Interchange of two adjacent letters	atlanta / altanta	2
MANY	Two or more of the same type or of different types	payed / paid	2-N

2) SELECTION OF CANDIDATE WORDS FOR CORRECTION CONSIDERING THE EDIT DISTANCE

For the selection of correction candidate words, it is necessary to identify the type of typographical error-related errors. In Table 2, there are cases in which the surroundings of the keyboard could not be pressed (XO, XY), the key was pressed twice too quickly (SWAP), the typing of a key was missed (OX, DOUB21), and a key was accidently pressed (DOUB12). The selection of correction candidate words must be considered by calculating the edit distance. The distribution of errors in general documents, except for spacing errors or apostrophe errors in the criteria divided into the spelling error detail classification, is as follows [18]: errors with an edit distance of 1 (OX, XO, DOUB12, DOUB21, XY) constitute 46.84% of the total number of errors; those with an edit distance of 2 (SWAP) constitute 22.78%; and those with an edit distance of 2 is not in the category of Table 2 or more (MANY) is 30.38%. This study attempts to correct errors in a wide range of documents (for all the aforementioned cases except MANY). However, for words that have lost more than half of their alphabets, it is difficult to infer the original word; thus, this study does not attempt to correct such words.

IV. CONTEXT-SENSITIVE SPELLING CORRECTION LANGUAGE MODEL

The context-sensitive spelling error problem considered in this study is solved based on the deep learning language model, which recently showed good performance in various tasks of natural language processing. The language model used in this study can be divided into three categories, shown in Table 3. First, an AR language model uses unidirectional (forward, backward) and bidirectional contextual information, and the vector expression of the word varies according to the surrounding context. This method has the advantage of reflecting contextual information in the word vector better than existing word embedding methods. However, the AR language model has limitations in bidirectional learning. Furthermore, it is not enough that it is a bidirectional learning model in the true sense because long short-term memory [19] learning performed in the forward and backward directions is independent, and the results are combined in the last layer in the pre-training process. AR language models include the embeddings from language model (ELMo) [20], generative pre-trained transformer (GPT) [21], and GPT-2 [16]. In this study, context-sensitive spelling error correction experiments were conducted using pre-training data provided by AllenNLP and OpenAI. The learning method of the AR language model can be explained using Equation (8). The probability $p(x)$ of the input sequence $(x_1, x_2, \cdots, x_T)$ is represented by the product of the conditional probability $p(x_t|x_{<t})$ in the forward (backward possible) direction. The model learns these conditional distributions as the objective. The negative-log probability of the AR model should be directional, and only unidirectional information is used. Therefore, it may be difficult to understand the sentence deeply using the bidirectional context.

$$\begin{aligned} &\textit{input sequence} : x = (x_1, x_2, \cdots, x_T) \\ &\textit{forward likelihood} : p(x) = \prod_{t=1}^{T} p(x_t \mid x_{<t}) \\ &\textit{training objective (forward)} : \\ &\max_{\theta} \log p_\theta(x) = \max_{\theta} \sum_{t=1}^{T} \log p(x_t|x_{<t}) \end{aligned} \quad (8)$$

The AE language model entails a technique of restoring input values. It focuses on matching the words that are masked to learn the process of restoring the words using noise (masking) in some of the sentences. The AE language model is primarily developed by applying various masking techniques, based on word restoration methods. It is also called the denoising auto-encoder. The AE language

TABLE 3. Classification and pre-training data information of language models used in the paper.

Model name	*Developer*	*Pre-training data volume*	*Embedding model classification*	
GloVe[13] (2014)	Stanford Univ.	600B tokens (Common Crawl)	word embedding	
fastText[14] (2017)	Facebook AI	840B tokens (Common Crawl)		
ELMo[20] (2018. 3)	AllenNLP	5.5B words (Wikipedia 1.9B + WMT 2008-2012 3.6B)	AR	contextual embedding
GPT[21] (2018)	OpenAI	800M words (Book corpus)		
GPT-2[16] (2019. 2)	OpenAI	40GB (8million web page data set)		
BERT[17] (2019. 1)	Google AI	16GB BERT data (Book corpus + Wikipedia 3.3 Billion words)	AE	
RoBERTa[22] (2019. 7)	Facebook AI	160GB (16 GB BERT data + 144GB additional)		
XLM-RoBERTa[23] (2019.11)	Facebook AI	2.5T (data across 100 languages)		
BART[24] (2019.10)	Facebook AI	160GB (16 GB BERT data + 144GB additional)		
T5[25] (2019.10)	Google	7T (C4/Colossal Clean Crawled Corpus dataset)		
XLNet[26] (2019. 6)	Google Brain	113GB (16GB BERT data + 97GB additional 33 Billion words)	Permutation	

models used in this study are BERT [17], the robustly optimized BERT approach (RoBERTa) [22], the cross-lingual language model RoBERTa (XLM-RoBERTa) [23], denoising sequence-to-sequence pre-training (BART) [24], and the text-to-text transfer transformer (T5) [25]. The context-sensitive spelling error correction experiment was conducted using pre-training data provided by Google AI and Facebook AI. The learning method of the AE language model can be explained with reference to Equation (9). We create a corrupted input that replaces the "[MASK]" token in the input sequence $(x_1, x_2, \cdots, x_T)$. The probability that the "[MASK]" token appears is not independent, but it is assumed to be independent. It is therefore represented by the product of each probability. The probability $p(\bar{x}|\hat{x})$ of the AE language model uses an objective function that maximizes it; $m_t = 1$ in the case of x_t being the "[MASK]" token and $m_t = 0$ in other cases. The objective function predicts only the "[MASK]" token using m_t. The AR language model has the advantage of using bidirectional self-attention to match the "[MASK]" token. However, there is a disadvantage that all "[MASK]" tokens are independently predicted via independency assumptions, and the dependency between them cannot be learned.

$$
\begin{aligned}
&\textit{input sequence} : \bar{x} = (x_1, x_2, \cdots, x_T) \\
&\textit{corrupted input} : \hat{x} = (x_1, \cdots, [MASK], \cdots, x_T) \\
&\textit{likelihood} : p\left(\bar{x} \mid \hat{x}\right) \approx \prod_{t=1}^{T} p\left(x_t \mid \hat{x}\right) \\
&\textit{training objective} : \\
&\max_{\theta} log\, p\left(\bar{x} \mid \hat{x}\right) = \max_{\theta} \sum_{t=1}^{T} m_t log\, p\left(x_t \mid \hat{x}\right) \qquad (9)
\end{aligned}
$$

The permutation language model was proposed to overcome the limitations of the AR and AE language models. It is a learning technique with a bidirectional learning effect because it learns unidirectionally using shuffled sentences. The permutation language model employed in this study is the generalized autoregressive pre-training model (XLNet) [26]. In this study, context-sensitive spelling error

TABLE 4. Scale of Google Web 1T.

	Text tokens	*3-gram*
Count	1,024,908,267,229	977,069,902

correction experiments were conducted using pre-training data, provided by the Google Brain team. The learning method of the permutation language model can be explained using Equation (10). A permutation is generated considering the index order of the input sequence $(x_1, x_2, \cdots, x_T)$, and the length of the sequence is the same as the factorial of T, i.e., the sequence x has the permutation of $T!$. The term Z_T is the set of all permutations of sequences with the length T, z_t is the t-th element, and $z_{<t}$ can be expressed as the objective when it is the $t-1$ element of permutation $z \in Z_T$. It is difficult to maximize the log probability in all permutations.

$$\begin{aligned} &\textit{input sequence} : x = (x_1, x_2, \cdots, x_T) \\ &\textit{likelihood} : \mathrm{E}_{z\sim Z_t}\left[\prod_{t=1}^{T} p\left(x_{z_t} \mid x_{z<t}\right)\right] \\ &\textit{training objective} : \max_{\theta} \mathrm{E}_{z\sim Z_t}\left[\sum_{t=1}^{T} log\, p_\theta\left(x_{z_t} \mid x_{z<t}\right)\right] \end{aligned} \tag{10}$$

V. EXPERIMENT

A. ERROR WORD GENERATION WITH CONTEXT REFERENCE

It can be difficult to obtain a large test corpus for experiments entailing context-sensitive spelling errors. The reason is that natural spelling errors should be collected while writing sentences, and if sentences that include these errors in small quantities are tested, it is difficult to measure a creditworthy performance for a wide range of words. Therefore, in this study, we create error words based on accurate sentences without errors, and we attempt to replace the error word with the sentence. In the method of generating error words, it is simple to replace the words of the sentence without errors using the edit distance. However, to measure the correction performance more reliably, the error word candidate is created, based on context information. Finally, the spelling error is reflected in the sentence.

To generate a large number of candidate error words that are similar to actual spelling errors, the Google Web 1T corpus with ten quadrillion word tokens is used, as shown in Table 4. The Google Web 1T corpus omits information with a frequency of 40 or less and is divided into an n-gram form, from 1-gram to 5-gram. Using the 3-gram from among the various n-grams, we attempt to generate error words. The reason for using 3-gram is that the number of candidate words is too high when using 2-gram, but too low when using 4-gram or higher. Referring to Equation (11), if an error candidate is found, it is an operation to find a word set of "*" that three 3-grams $(w_{i-2}\ w_{i-1}\ *)$, $(w_{i-1}\ *\ w_{i+1})$, and $(*\ w_{i+1}\ w_{i+2})$ commonly contain. The term $(w_{i-2}\ w_{i-1}\ *)$ has the word and frequency of the third position of all 3-grams starting with "$w_{i-2}\ w_{i-1}$". Based on this operation, the result of obtaining all the words that can appear in "*," the position of the error generation candidate word, is called the candidate lexicon (CL), defined in Equation (11).

$$\begin{aligned} CL = &< w_{i-2}, w_{i-1}, * > \cup < w_{i-1}, *, w_{i+1} > \\ &\cup < *, w_{i+1}, w_{i+2} > \end{aligned} \tag{11}$$

Figure 6 presents a schematic of the error candidate search method, defined in Equation (11), showing the process of finding all "*" that satisfy (a b *) ∪ (b * c) ∪ (* c d) in the sentence "a b * c d". In the frequency dictionary of 3-gram extracted from the corpus, three candidates ("a b e", "a b w", and "a b b") satisfying "a b *," two candidates ("b v c" and "b q c") satisfying "b * c," and two candidates ("q c d" and "e c d") satisfying "* c d" can be found. If the search 3-gram is sorted through a combination operation, the duplicated word is removed and the final word "e, w, b, v, q" corresponding to "*" is obtained. In this process, the total number of pre-accesses for the search of the total 3-gram is 7, and the total number of final words extracted through the combined operation is 5.

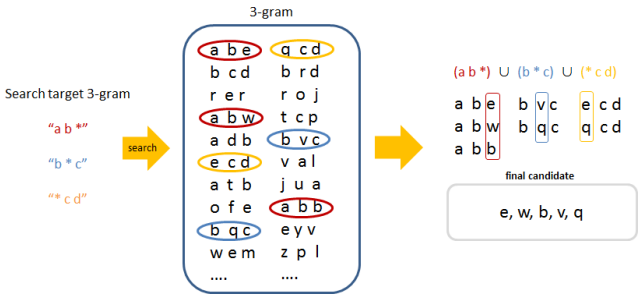


FIGURE 6. Union calculation in Default operation.

As in Figure 6, finding a candidate set for "*" does not complete the error candidate set. Filtering is required to find the error candidate. The criteria are described below. In the correct sentence, the edit distance between the word of the candidate word set and the error generation target word (position in the place of "*") is calculated, and the word corresponding to the closet distance is selected. For example, when an edit distance of 2 (below 2) is used for the error-generating target word "Wors," as shown in Table 5, the candidate sets will have a set of error words such as "world," "ears," and "work." The error is classified based on the obtained candidate error word set, as shown in Table 2. An error word can randomly be selected from the obtained candidate error word set or using sentence probability with reference to frequency information. From the frequency information in Table 5, it can be observed that it is difficult for a 3-gram in three directions to appear in duplicate in the corpus of ten quadrillion words. The error candidate also includes words that are not in the dictionary. In fact, this is an error extracted from the 3-gram, obtained for a frequency of 40 or more, and it can be considered as a spelling error that users often commit in a keyboard input environment. This method of

TABLE 5. Produce a Candidate Word in 3-gram-based Context.

Example sentence			
… for open wors(correction target) with statistical …			
Error word candidate	*3-gram*		
	Left	*middle*	*right*
fora	0	40	0
world	618	190	0
work	237	200	669
form	58	691	0
lots	265	153	0
wire	830	40	0
warr	63	0	0
jobs	930	145	0
toes	249	0	0
rods	103	0	0
port	303	813	0
ears	284	0	0
cows	49	0	0
ward	0	89	0
cars	169	148	0
bars	77	72	0
word	0	178	0
sore	0	217	0
more	0	73	0
jars	0	72	0
burs	0	41	0
works	0	0	63
loss	0	0	126

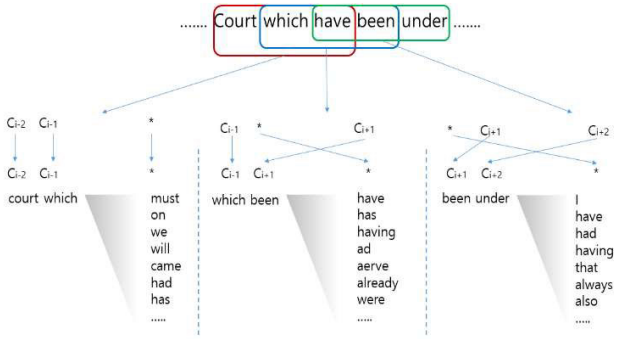


FIGURE 7. Refined 3-gram information in search.

error generation will measure the reliability of results in the experiment of context-sensitive spelling error correction.

As shown in Table 4, the 3-gram extracted from the corpus of ten quadrillion words contains approximately 1 billion data points, and the search cost of the total 3-gram is significantly high in the error set generation process. To overcome this limitation, the experiment in this study was performed by changing the structure to be more convenient for searching, as shown in Figure 7.

Figure 7 shows an example of the data structure constructed for the "*" search operation. The number of 3-grams

TABLE 6. Produce a candidate word in 3-gram-based context.

	Basic	*Left*	*Middle*	*Right*
3-gram	60GB	10.9GB	11.9GB	11.9GB
		sum : 34.7GB		

TABLE 7. Comparison of 3-gram search speed using Google LevelDB[27].

	Basic search speed	*Improve structure speed*
Document full search time(sec)	96	4.6
Search time per word(sec)	0.00169	0.00008

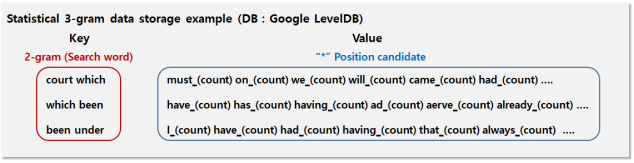


FIGURE 8. Information on candidate words configured in one 3-gram.

extracted from the Google Web 1T corpus is very large; therefore, a sequential search of statistical information is time intensive. To address this issue, we propose a simple high-speed search method. First, to explain the 3-gram "Court which has" in Figure 7, we divide the three words into three parts because we do not know which of the three words will correspond to "*". In the case of "Court which have", the possible positions are

"(*Court) which have"
"Court (*which) have"
"Court which (*have)."

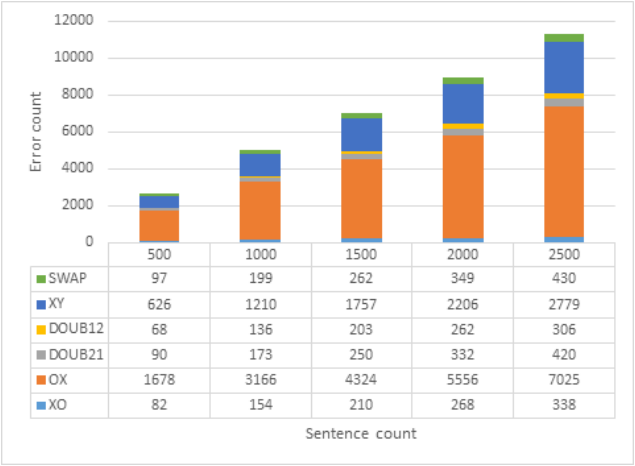
In the next step, each word in the "∗" position is moved to the right.

"(*Court) which have" → "which have (*Court)"
"Court (*which) have" → "Court have (*which)"
"Court which (*have)" → "Court which (*have)"

When data is obtained by the "*" included in the whole 3-gram, it can be observed that various words, shown in Figure 8, are in co-occurrence. If data is stored in this way, not only can the storage data be reduced, as in Table 6, but the search time can also be reduced. Table 6 shows that the storage capacity has decreased by approximately 42%. There is also a significant difference in the search speed, shown in Table 7.

1) ERROR WORD GENERATION IN TEST DOCUMENTATION

Figure 9 displays a frequency graph of error words extracted from the Brown Corpus, based on the error types in Table 2. The context information was referenced using 3-grams, and each error word is the actual word shown in context. The



	500	1000	1500	2000	2500
SWAP	97	199	262	349	430
XY	626	1210	1757	2206	2779
DOUB12	68	136	203	262	306
DOUB21	90	173	250	332	420
OX	1678	3166	4324	5556	7025
XO	82	154	210	268	338

FIGURE 9. Increment graph by error type, according to the increase of error generation document.

The City Purchasing Department, the jury said, "is lacking in experience clerical personnel as a resalt of city personnel policies". It urge that the cit "take steps to remedy" this problem.
It urged hat the nxt Legislature "provide enabling finds and re-set the effective ate so that an orderly implementaton of the law may be effected".
The grand jury took a wipe at the tate Welfare Department's handling of federal fund granted for child welfare service in foster homes.

FIGURE 10. Examples of errors generated in actual documents (OX).

information of the total error words was searched, based on Google Web 1T 3-gram, and the error types of OX and XY were observed to be most frequent. Figure 10 shows examples of errors of the OX type in Brown Corpus; the red words correspond to error words. The error words used in the experiment are searched for all the words that can appear in the target position through the search of 3-gram, selected by considering the edit distance, and selected the sentence and the probabilistic high word among the several error candidates. Each error word is assumed to have occurred independently, except when two or more occur simultaneously.

B. PERFORMANCE INDICATORS IN CONTEXT-SENSITIVE SPELLING CORRECTION EXPERIMENT

The performance measurement criteria of the context-sensitive spelling error detection and correction experiment are divided into precision and recall, respectively, as shown in Equation (12). Precision and recall are not different from the denominator values in the conventional equation; however, the value of the numerators is different. The numerators in the detection equation apply all cases where they are replaced by other candidates through probability value comparison, and the numerators in the correction equation apply the correct answer by selecting it from the values obtained from the detection.

$$\begin{aligned} Precision\,(Detection) &= \frac{Corrected\ word\ of\ count}{count\ of\ word\ judged\ to\ be\ error} \\ Precision\,(Correction) &= \frac{Correctly\ corrected\ word\ count}{count\ of\ word\ judged\ to\ be\ error} \\ Recall\,(Detection) &= \frac{Corrected\ word\ of\ count}{Error\ word\ of\ total\ count} \\ Recall\,(Correction) &= \frac{Correctly\ corrected\ word\ count}{Error\ word\ of\ total\ count} \end{aligned} \tag{12}$$

The F-measure (or F1-score) can be used to represent more simply the previously obtained equations. The F-measure is also called a harmonious mean because it overcomes the imbalance of data and processes values with balanced data to calculate and adjust the same case in all cases. Because the precision and recall obtained under different conditions are unbalanced, the harmonic mean, which gives uniformity to the performance value, is highly reliable. The F-measure is expressed by Equation (13).

$$F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{13}$$

C. COMPARISON OF EMBEDDING-BASED CORRECTION PERFORMANCE

In this subsection, we compare the embedding-based correction performance. The word embedding language models used in this study are two language models, GloVe and fastText, mentioned in Table 3. The other language model corresponds to a contextual embedding language model, in which embedding information changes according to contextual information.

In the experiment, the statistical model was used to search for errors and the creation of candidate words, and the embedding information of each language model was used for correction. Pre-training data, provided by the researchers of each language model, were used. The context-dependent spelling error correction test was performed using the Brown Corpus, a balanced corpus constructed by a specialist. This test was performed on 930 randomly selected sentences (100,557 words / MS word document with 45 pages). The error words in the experiment were generated according to the error types in Table 2, and the edit distance used in the correction model for the correction calculation was not limited.

First, the performance presented in Table 8 and Figure 11 was obtained by employing a contextual information left/right window size of 10, based on the target correction word. As a result, most sentence lengths do not exceed 21 words; therefore, the entire sentence is referred. Figure 11 shows how the performance changes as the window size of contextual information increases. The reason the overall prediction detection is higher than the other experimental values in Table 8 is owing to the performance of the statistical language model responsible for the function

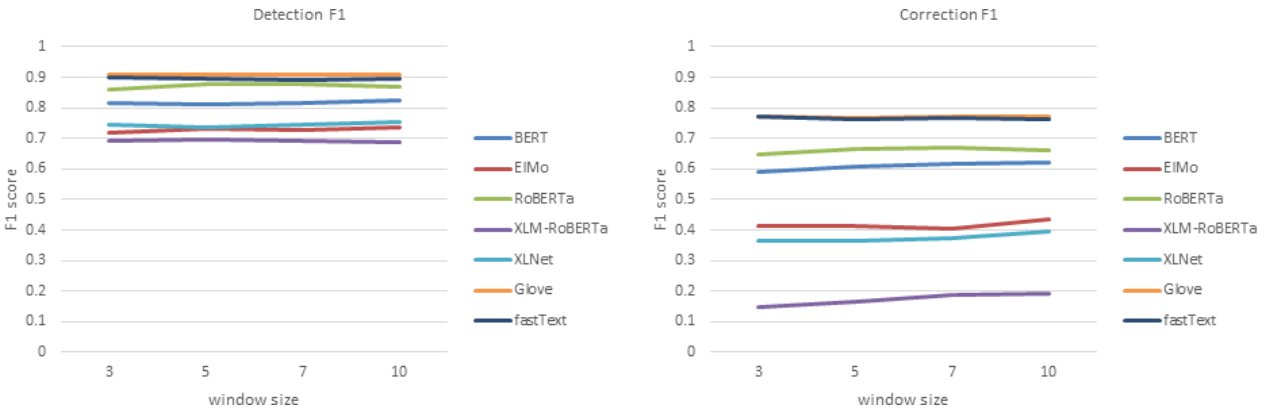


FIGURE 11. Comparison of context-dependent spelling correction performance, based on word embedding language model.

TABLE 8. Comparison of embedding based correction performance (precision, recall, F1).

		Detection	*Correction*
BERT	***Precision***	98.32%	74.03%
	Recall	71.19%	53.60%
	F1	82.58%	62.18%
ELMo	***Precision***	97.98%	58.07%
	Recall	59.02%	34.98%
	F1	73.67%	43.36%
RoBERTa	***Precision***	98.47%	74.48%
	Recall	78.11%	59.40%
	F1	87.11%	66.25%
XLM-RoBERTa	***Precision***	97.75%	27.16%
	Recall	52.85%	14.68%
	F1	68.61%	19.06%
XLNet	***Precision***	98.05%	51.34%
	Recall	61.27%	32.09%
	F1	75.41%	39.49%
GloVe	***Precision***	98.57%	83.75%
	Recall	84.00%	71.38%
	F1	90.70%	77.07%
fastText	***Precision***	98.54%	83.54%
	Recall	82.32%	69.97%
	F1	89.70%	76.25%

TABLE 9. Comparison of AR and AE language model based correction performance (precision, recall, F1).

		Detection	*Correction*
RoBERTa	***Precision***	**96.74%**	**96.07%**
	Recall	**95.80%**	**95.13%**
	F1	**96.27%**	**95.60%**
BART	***Precision***	96.67%	93.45%
	Recall	93.56%	90.44%
	F1	95.09%	91.92%
BERT	***Precision***	95.54%	79.42%
	Recall	69.03%	57.38%
	F1	80.15%	66.63%
GPT	***Precision***	96.09%	89.45%
	Recall	79.13%	73.66%
	F1	86.79%	80.79%
GPT2	***Precision***	96.44%	91.43%
	Recall	87.21%	82.68%
	F1	91.59%	86.84%
XLM-RoBERTa	***Precision***	96.68%	85.26%
	Recall	93.79%	82.72%
	F1	95.21%	83.97%
T5	***Precision***	96.17%	91.02%
	Recall	80.84%	76.51%
	F1	87.84%	83.14%
XLNet	***Precision***	94.43%	89.57%
	Recall	54.66%	51.85%
	F1	69.25%	65.67%

of the error search. Even if the statistical language model correctly judged that an instance may correspond to an error, the embedding-based correction method often failed to make the final correction; thus, the other values are low. The F1 values of detection and correction in Figure 11 show that the word embedding techniques, GloVe and fastText, show higher experimental performance than the contextual embedding language models in context-sensitive spelling error correction, based on embedding information. Among them, XLM-RoBERTa is a metric model based on RoBERTa; however, its performance is significantly lower than that of RoBERTa. This is owing to the high level of noise generated by learning 100 languages for translation. By observing the overall results, we can confirm that performing corrections via the comparison of words with words results in a superior correction performance, based on embedding information.

D. COMPARISON OF CORRECTION PERFORMANCE BASED ON AR AND AE LANGUAGE MODELS

In this subsection, we compare the performance of context-sensitive spelling error correction, based on AR and AE language models. The experimental conditions are the same as those for the correction based on embedding, in Section 5C.

TABLE 10. Detailed correction performance based on AR and AE language models.

SWAP	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	**99%**	**99%**	**99%**	**99%**	**99%**	**99%**
BART	99%	97%	98%	99%	97%	98%
BERT	99%	68%	80%	79%	54%	64%
GPT	99%	87%	93%	99%	87%	93%
GPT2	99%	93%	96%	98%	92%	95%
T5	99%	97%	98%	90%	88%	89%
XLM-RoBERTa	99%	88%	93%	95%	85%	90%
XLNet	99%	81%	89%	98%	81%	88%

XY	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	**97%**	**94%**	**95%**	**96%**	**93%**	**94%**
BART	97%	91%	94%	94%	88%	91%
BERT	96%	67%	79%	80%	56%	66%
GPT	96%	77%	86%	90%	72%	80%
GPT2	97%	83%	89%	91%	78%	84%
T5	97%	94%	95%	83%	80%	82%
XLM-RoBERTa	96%	77%	85%	92%	73%	81%
XLNet	94%	46%	62%	87%	42%	57%

DOUB12	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	**100%**	**100%**	**100%**	**98%**	**98%**	**98%**
BART	100%	97%	98%	98%	95%	97%
BERT	100%	77%	87%	78%	60%	68%
GPT	100%	82%	90%	96%	78%	86%
GPT2	100%	90%	95%	96%	87%	91%
T5	100%	93%	97%	86%	80%	83%
XLM-RoBERTa	100%	93%	97%	86%	80%	83%
XLNet	100%	75%	86%	91%	68%	78%

DOUB21	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	99%	96%	97%	**99%**	**96%**	**97%**
BART	99%	94%	96%	99%	94%	96%
BERT	99%	76%	86%	91%	70%	79%
GPT	99%	85%	92%	99%	85%	92%
GPT2	99%	90%	94%	99%	90%	94%
T5	**99%**	**96%**	**98%**	87%	85%	86%
XLM-RoBERTa	99%	82%	90%	94%	78%	85%
XLNet	99%	55%	71%	95%	54%	69%

OX	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	**96%**	**96%**	**96%**	**96%**	**95%**	**96%**
BART	96%	94%	95%	92%	90%	91%
BERT	95%	69%	80%	79%	57%	66%
GPT	95%	79%	86%	88%	73%	80%
GPT2	96%	88%	92%	90%	83%	86%
T5	96%	93%	95%	85%	83%	84%
XLM-RoBERTa	96%	81%	87%	90%	76%	83%
XLNet	94%	55%	69%	89%	52%	66%

XO	Detection			Correction		
	Precision	Recall	F1	Precision	Recall	F1
RoBERTa	**98%**	**97%**	**97%**	**98%**	**97%**	**97%**
BART	98%	93%	95%	95%	90%	93%
BERT	97%	71%	82%	76%	56%	65%
GPT	97%	77%	86%	91%	72%	81%
GPT2	97%	88%	93%	95%	86%	90%
T5	98%	94%	96%	91%	87%	89%
XLM-RoBERTa	97%	91%	94%	95%	89%	92%
XLNet	96%	60%	74%	91%	57%	70%

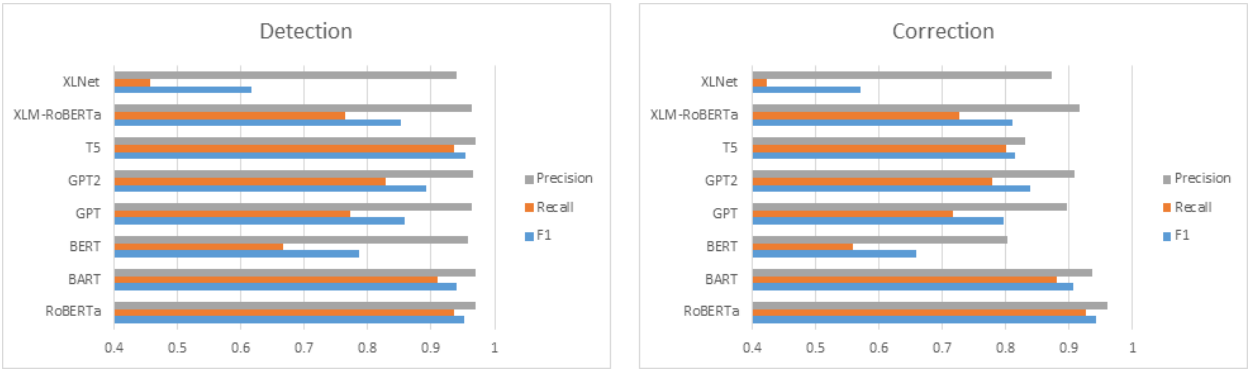


FIGURE 12. Comparison of context-sensitive spelling error correction performance, based on AR and AE language models.

The error word detection is performed by the statistical language model, and the deep learning-based language model is responsible for the correction. As with the embedding-based correction experiment, the training data of the language model used for correction were pre-training data provided by researchers of each language model. Table 9 shows the result of correcting the context information left/right window size of 10 based on the target correction word.

The F1 performance obtained for detection and correction in this case is significantly higher than that of the embedding-based correction method elucidated in Section 5C. As observed from Figure 12, the method of correcting using the permutation language model resulted in the lowest performance. The permutation language model learns sentences by shuffling, and it is advantageous that the bidirectional sentence information can be referred to in the learning method of the AR language model. For context-sensitive spelling error correction, it is determined that the learning method that employs sentence shuffle entails noise that reduces correction performance. BART and RoBERTa, which resulted in a high performance, show that the AE language model is the most suitable for context-sensitive spelling error correction. The most significant reason for this is that learning creates a random mask in a sentence and restores it by referring to a bidirectional context. The GPT-2 performance is lower than that of BART and RoBERTa; however, the AR language model shows good correction performance. The AR language model learns by predicting the next word of the sentence. The performance is lower than that of the learning method of the AE language model, as it refers to the unidirectional context. GPT and BERT, which are early models of the AR and AE language models, generally have a lower performance than the metric model. BERT uses the word piece model of tokenizer. therefore, it is robust for OOV cases, but shows poor performance in the correction of error words. XLM-RoBERTa, a metric model of RoBERTa, is considered to have a lower correction performance than other BERT metric models, owing to noise generated by learning multiple languages for translation. T5, which is a learning method specializing in the fill-in-the-blank operation, showed good detection performance; however, it was confirmed that the correction performance was lower than that of other BERT metric models. The most significant reason for the decreased performance of T5 learning using sentinel tokens, which function as a large mask in one sentence, is that it is vulnerable to correcting spelling errors where only one correction word is observed because it can be applied to several words where sentinel tokens are gathered. Table 10 subdivides the performances displayed in Table 9 according to the error types presented in Table 2, based on the AR and AE language models.

VI. CONCLUSION

In this study, we applied various deep learning language models to correct context-sensitive spelling errors. The results show that the correction of context-sensitive spelling errors accounts for the detection and correction of more than 96%(F1) of errors. In this paper, we propose an approach to correcting various context-sensitive spelling errors based on deep learning. This includes a basic correction method that employs the distance value in word-to-word learning in a unidirectional context, a correction method that employs the AR language model, which predicts the next word through the uni-directional context, and a correction method that employs. the AE language model, which restores the word using bidirectional context information. Correcting spelling errors is one of the functions of a spell checker, and the problem of other spelling or spacing errors in sentences must be addressed in the future. This research will therefore continue for as long as humans use language.

REFERENCES

[1] C. W. Young, C. M. Eastman, and R. L. Oakman, "An analysis of ill-formed input in natural language queries to document retrieval systems," *Inf. Process. Manage.*, vol. 27, no. 6, pp. 615–622, Jan. 1991.

[2] A. M. Wing, and A. D. Baddeley, "Spelling errors in handwriting: A corpus and distributional analysis," in *Cognitive Processes in Spelling*, London, U.K.: Academic, 1980, pp. 251–285.

[3] K. W. Church and W. A. Gale, "Probability scoring for spelling correction," *Statist. Comput.*, vol. 1, no. 2, pp. 93–103, Dec. 1991.

[4] M. Kim, S.-K. Choi, J. Jin, and H.-C. Kwon, "Adaptive context-sensitive spelling error correction techniques for the extremely unpredictable error generating language environments," (in korea), in *Proc. IEEE Int. Conf. Comput. Inf. Technol.; Ubiquitous Comput. Commun.; Dependable, Autonomic Secure Comput.; Pervasive Intell. Comput.*, Oct. 2015, pp. 654–656.

[5] J.-H. Lee, M. Kim, and H.-C. Kwon, "Improved statistical language model for context-sensitive spelling error candidates," (in korea), *J. Korea Multimedia Soc.*, vol. 20, no. 2, pp. 371–381, Feb. 2017.
[6] E. Mays, F. J. Damerau, and R. L. Mercer, "Context based spelling correction," *Inf. Process. Manage.*, vol. 27, no. 5, pp. 517–522, Jan. 1991.
[7] M. Kim, H.-C. Kwon, and S. Choi, "Context-sensitive spelling error correction using Eojeol N-gram," (in korea), *J. KIISE*, vol. 41, no. 12, pp. 1081–1089, Dec. 2014.
[8] H. Li, Y. Wang, X. Liu, Z. Sheng, and S. Wei, "Spelling error correction using a nested RNN model and pseudo training data," 2018, *arXiv:1811.00238*. [Online]. Available: http://arxiv.org/abs/1811.00238
[9] H. Gong, Y. Li, S. Bhat, and P. Viswanath, "Context-sensitive malicious spelling error correction," 2019, *arXiv:1901.07688*. [Online]. Available: http://arxiv.org/abs/1901.07688
[10] J.-H. Lee, M. Kim, and H.-C. Kwon, "Context-sensitive spelling error correction techniques using word embedding," (in korea), in *Proc. KIISE Korea Comput. Congr.*, 2018, pp. 607–609.
[11] J.-H. Lee, M. Kim, and H.-C. Kwon, "Context-sensitive spelling error correction techniques using contextual embeddings," (in korea), in *Proc. KIISE Korea Comput. Congr.*, 2019, pp. 497–499.
[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781
[13] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[14] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," 2016, *arXiv:1607.04606*. [Online]. Available: http://arxiv.org/abs/1607.04606
[15] T. Brants and A. Franz, "Web 1T 5-gram corpus version 1.1," Google Res., Mountain View, CA, USA, Tech. Rep., Feb. 2020. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2006T13
[16] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805
[18] Y. Bestgen and S. Granger, "Categorising spelling errors to assess l2 writing," *Int. J. Continuing Eng. Edu. Life-Long Learn.*, vol. 21, nos. 2–3, pp. 235–252, 2011.
[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[20] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018, *arXiv:1802.05365*. [Online]. Available: http://arxiv.org/abs/1802.05365
[21] A. Radfor *et al.* (May 1, 2020), *Improving Language Understanding by Generative Pre-Training*. [Online]. Available: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf
[22] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*. [Online]. Available: http://arxiv.org/abs/1907.11692
[23] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," 2019, *arXiv:1911.02116*. [Online]. Available: http://arxiv.org/abs/1911.02116
[24] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019, *arXiv:1910.13461*. [Online]. Available: http://arxiv.org/abs/1910.13461
[25] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified Text-to-Text transformer," 2019, *arXiv:1910.10683*. [Online]. Available: http://arxiv.org/abs/1910.10683
[26] Z. Yang, Z. Dai, Y. Yang. J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 5753–5763.
[27] S. Ghemawat and J. Dean, "LevelDB," Google Inc., Mountain View, CA, USA, Tech. Rep., Feb. 2020, vol. 70. [Online]. Available: https://github.com/google/leveldb

JUNG-HUN LEE received the M.S. degree from the Department of Computer Science and Engineering, Pusan National University, Busan, South Korea, in 2017, where he is currently pursuing the Ph.D. degree.

His current research interests include natural language processing, machine learning, knowledge representation, and its applications.

MINHO KIM received the M.S. and Ph.D. degrees from the Department of Computer Science and Engineering, Pusan National University, Busan, South Korea, in 2007 and 2011, respectively.

He has been a Professor with the Catholic University of Pusan, Busan, since 2020. His current research interests include natural language processing, information retrieval, machine learning, and artificial intelligence.

HYUK-CHUL KWON received the M.S. and Ph.D. degrees in computer engineering from Seoul National University, Seoul, South Korea, in 1984 and 1987, respectively.

He has been a Professor with Pusan National University, Busan, South Korea, since 1988. He was a Visiting Professor and a Researcher with CSLI, Stanford University, Stanford, CA, USA, from 1992 to 1993. He has served as a Consultant at the Xerox Palo Alto Research Center, Palo Alto, CA, USA, in 1993. He is currently the Head of the School of Electrical and Computer Engineering and the Director of the Specialized Group of Industrial Automation, Information, and Communication, Pusan National University. His research interests include natural language processing, information retrieval, machine learning, and its application.

Dr. Kwon is a member of ACM.