Profane or Not: Improving Korean Profane Detection using Deep Learning

Jiyoung Woo[1], Sung Hee Park[2], and Huy Kang Kim[3,*]
[1] Big Data Engineering Department, Soonchunhyang University
22, Soonchunhyang-ro, Sinchang-myeon, Asan-si, Chungcheongnam-do, Republic of Korea
[e-mail: jywoo@sch.ac.kr]
[2] Hanwha Investment & Securities Co
56, Yeoui-daero, Yeongdeungpo-gu, Seoul, Republic of Korea
[e-mail: sunghee.park@hanwha.com]
[3] Graduate School of Information Security, Korea University
145, Anam-ro, Seongbuk-gu, Seoul, Republic of Korea
[e-mail: cenda@korea.ac.kr]
[*]Corresponding author: Huy Kang Kim

Received November 8, 2021; accepted December 14, 2021;
published January 31, 2022

Abstract

Abusive behaviors have become a common issue in many online social media platforms. Profanity is common form of abusive behavior in online. Social media platforms operate the filtering system using popular profanity words lists, but this method has drawbacks that it can be bypassed using an altered form and it can detect normal sentences as profanity. Especially in Korean language, the syllable is composed of graphemes and words are composed of multiple syllables, it can be decomposed into graphemes without impairing the transmission of meaning, and the form of a profane word can be seen as a different meaning in a sentence. This work focuses on the problem of filtering system mis-detecting normal phrases with profane phrases. For that, we proposed the deep learning-based framework including grapheme and syllable separation-based word embedding and appropriate CNN structure. The proposed model was evaluated on the chatting contents from the one of the famous online games in South Korea and generated 90.4% accuracy.

Keywords: Profanity, deep learning, convolutional neural network, text mining, natural language processing.

1. Introduction

Profanity among users of online PC games has been an issue since the late 1990s, when PC rooms (i.e., Netcafé) first appeared. Now that the Korean game industry has matured, abusive language that includes profanities is still encountered in online PC gaming, mobile gaming, chatting, etc. To address this issue, the game industry has taken various measures, such developing abusive language prevention systems and including them in games, disciplining gamers with notices, and introducing education programs.

To prevent profanity in games, game companies have created systems in which certain words are replaced by strings of asterisks when they are entered in gaming chat rooms. For example, if "개[…]" (i.e., dog- in English) or "[…]새끼" (i.e., -son in English) is typed in a chat prompt, it will appear as "***." However, this has not eliminated profanity in chat windows because gamers use special characters or subtly change a word's connotation to avoid using the original profane words, but the reader would understand the implied profanity [1]. According to an industry representative, "Game chat rooms are equipped with profanity filtering systems, but it is not easy to deal with users who employ special characters to create unusual neologisms." If profanity is severe, gamers may quit the game. In extreme cases, it may lead to slander and become a legal issue. Therefore, restricting profanity is an important issue for game companies [2]. Consequently, game companies currently use a process in which a game master receives a profanity notice and personally reads it to make a judgment, which takes considerable time. Moreover, game masters are continuously exposed to abusive language, and this creates psychological stress.

So far, most profanity detection systems have been based on lists of prohibited words. However, systems that are based on prohibited words have faced several problems as pointed out in Sood and Churchill's work [3]. In their works, the filtering system using popular profanity words lists showed low precision and low recall in detecting profanity. First, they are inefficient in detecting when profanity is used in an altered form. Owing to the characteristics of Korean characters that have various pronunciations, words can be changed into different forms that have similar pronunciations. Additionally, due to the internet's unique decomposition of the Korean language, filtering functions can be bypassed by combining special characters that have similar shapes, Chinese characters, and foreign words. Second, there are cases where otherwise normal words are considered profane because the word is transformed while the proposition is attached. For example, if the text "처음부터 가능해요" (i.e., It's ok from the first time) is entered in the chat prompt, the word "음부" (i.e., pubic) is filtered, making it difficult to have an ordinary conversation. Owing to such cases, it is difficult to determine the scope of prohibited words. Third, due to the characteristics of the Korean script, spacing is inconsistent, and this adds to the difficulties experienced by filtering systems. For example, "못찾으니미치겠네"(i.e 'm going crazy because I can't find it) contains profane word "니미"(i.e. I will kill your mom), but this word is formed by miss spacing. Fourth, Korean basically allowes different meaning for a word. Even in the phrase "Channel 18," the "18" can refer to the profanity "시발." (i.e., shit). Thus, word filtering or machine learning using morpheme-based separation is not working for Korean language.

Previous works were proposed to contribute to building the profanity detection system including altered form. However, to our best knowledge, few study deals with the drawback that normal words are detected as profanity. This study aims to provide a way to recognize real profanity from filtered sentences in a dictionary of prohibited words using a convolutional neural network (CNN). Deep learning automatically learns features from text, and this can reduce the burden on game masters.

2. Related Works

2.1 Profanity Detection

Most studies on profanity utilize language-based machine learning models; however, the distinguishing factors are the features used in the models. For instance, in Chen's et al.'s work [4], "lexical syntactic" features were used in naïve Bayes and support vector machine (SVM) systems to classify malicious replies on YouTube. In Nobata at el.'s study [5], Yahoo finance and news articles were used to classify profanity based on Vowpal Wabbit's regression model. N-grams, linguistic features, syntactic features, lexicons, and embedding-derived features were combined to increase accuracy. There have also been studies on profanity filtering systems that use N-gram-based SVMs to distinguish profanities without managing a list of prohibited words [6]. Lee et al. [7] designed a decision system that successfully detects (obfuscated) abusive text using an unsupervised learning of abusive words based on word2vec's skip-gram and the cosine similarity. Their system also deploys several efficient gadgets for filtering abusive text such as blacklists, n-grams, edit-distance metrics, mixed languages, abbreviations, punctuation, and words with special characters to detect the intentional obfuscation of abusive words. Martens et al. [8] developed a system that classifies words into several categories and annotates them to understand toxicity in games.

Ratadiya and Mishra [9] pointed that profane text can be categorized into multiple categories like toxic, hate, insult, etc. and interest has now grown beyond binary categorization. They proposed the use of attention mechanism in recurrent neural network. Founta et al. [10] proposed a deep learning architecture, which utilizes the meta data such as the number of followers, the location, account age, total number of liked tweets, and combines it with automatically-extracted hidden patterns within the text of the tweets, to detect multiple abusive behavioral norms. This work used RNN for text processing. Yenala et al. [11] proposed that convolutional Bi-Directional LSTM which combines the strengths of both convolution neural networks and bi-directional LSTMs (BLSTM) for profanity detection in user conversations in messengers.

Previous works mostly focus on English profanity, which can be handled with matching and replacing terms with a list of offensive words and their variants. However, there are times when words of praise are used sarcastically. This problem is challenging for all languages. Other languages except English face other problems. A Chinese character appearing in an offensive word may also appear in a mild word [12]. In Korean language, the syllable is composed of graphemes and words are composed of multiple syllables, it can be decomposed into graphemes without impairing the transmission of meaning, and the form of a profane word can be seen as a different meaning in a sentence. The detection of profanity in Korean language is challenging. Some Korean researchers performed related works [13,14]. However, they missed the point that filtering system can detect normal words as profane words.

2.2 Machine Learning for Text Classification

In machine learning, the algorithm performance is affected by the specific features extracted from the data. The disadvantage is that the model built from specific features cannot be deployed in other datasets or other domains. Consequently, deep learning models have recently emerged as an alternative to limited machine learning methods because they can define arbitrary features. Recursive neural networks (RNNs) are effective at analyzing text because they learn in a manner that saves the word order. Likewise, convolutional neural networks (CNNs) use convolution filters to discern continuous patterns, and they learn efficiently compared to RNNs, even when the amount of data is small. Currently, CNNs are attracting attention in the field of natural language processing. Kim [15] used CNNs to achieve excellent performance in various text classification tasks, and this has become the standard for new text classification architectures.

A neural network model that uses continuous BOW (CBOW) and paragraph2vec to detect profanity has also been constructed in Djuric at el.'s work [16]. A transfer learning method was further proposed, in which profane and normal text was learned by a CNN-based classifier, and adjustments were made to allow the neural network to classify features of profanity. Subsequently, in Seo and Cho's work [17], the filter was reused while learning the actual training text. Additionally, malicious replies in Wikipedia discussion pages were used to test an RNN with long short-term memory (LSTM) and word embedding, a CNN with word embedding, and a CNN with character embedding. Jue, Chu and Wang [18] found that the CNN with character-level embedding had the highest accuracy at 94%.

2.3 Text mining for Korean Language

In Korean language, simple symbols are matched for each of the phonemes. Koreans have difference from English in that they are grouping the symbols into a square block, a syllable. The first element in the block is the initial consonant and what follows usually to the let is the vowel nucleus. In addition, sometimes, coda follows below the consonant and vowel nucleus.

The significant difference of Korean from English is that morphemes may change shape depending on the preceding sounds. This makes to apply English-based language model to Korean difficult. Thus, some works are proposed by using deep learning models for Korean processing.

Kim, Ra and Lim [19] proposed BERT model that solves the task of zero anaphora resolution (omission of subject in the sentence) for completely understanding the texts in Korean.

Song, Park, and Shin [20] emphasized that Korean is a morphologically rich language, so it is difficult to apply English-based models. They develop a method of sentiment lexicon embedding with joint encoding morphemes and their POS tags for Korean. This is quite useful for Korean in which nouns or verbs are combined with a participle or with a tense. They developed an attention-based LSTM model for sentiment classification.

Nguyen et al. [21] built a lexical network to resolve the disambiguation of Korean, same pronunciation on different meaning words. This will be helpful to improve the deep learning-based model by inferring exact meaning of a word from the lexical network.

Lee, Kwon, and Ahn [22] proposed to use phoneme(grapheme) and built a RNN for Korean sentiment. Their results showed that phoneme-level which is smaller than morpheme-level outperforms the morpheme-level method. In Korean, phoneme is the smallest particles for sound and morpheme is the smallest particles for written languages. This work showed that Korean is morphemically diverse, so phoneme-level or morpheme-level is appropriate for natural language process.

3. Material and Methods

3.1 Data

The data used in this study were from the 5th closed beta service of ArcheAge, a massive multiplayer online role-playing game (MMORPG) developed by XLGames; the data were provided before regular service began. XLGames maintain the prohibited word dictionary to filter out profanity and game administrators double check the filtered chat.

The number of active members was at least 150,000. Only the chat content was extracted from 65,499 data items from 00:33:32 - 21:13:18 on August 15, 2012. We randomly selected 5,000 items that were considered to be profanity based on a prohibited word dictionary with 1,959 terms. These were manually examined and tagged as "profanity" or "not profanity." The monitoring criteria were dictionary-based. The profanity dictionary was provided by the game company, and if the words in this dictionary appeared in the chat, they were filtered as profanity. However, due to the nature of the Korean language, certain phrases have the same text as profanities but have different meanings or are contextually not profanities. To perform filtering, 5,000 filtered phrases were extracted through random sampling, and a person determined the meanings of these phrases and tagged them as profanity or not profanity. In total, 1,396 sentences were normal words that were incorrectly filtered as profanity from the prohibited word dictionary. Approximately 28% of the text can be incorrectly filtered as profanity through the dictionary-based method even though these are normal words, and this will be a burden to game administrators. To build the detection model**,** out of the 5,000 data items, 2,812 were used as training data, 1,250 were used as verification data, and 938 were used as test data.

Table 1. Status of the data used in the study

Class	Human-judged Profanity	Human-judged Normal Words	Total
Dictionary-based Profanity	3,604	1,396	5,000

Table 2 contains examples of normal text that may be judged as profanity based on a prohibited word dictionary. We categorized the causes of mis classification into same pronunciation but different meaning, same pronunciated syllables, and same pronunciation from wrong punctuation. Because Korean is a syllable language, some of the words may have the same pronunciation as other words. In addition, since many Korean words are derived from Chinese characters, there are also words with the same pronunciation but different meanings.

Table 2. Examples of normal words considered profanity based on a dictionary

Dictionary	Chatting Phrases (Korean, English)	Cause of mis classification
사정(cumshot)	*사정*을 여쭤보니 …(When I asked about the situation…).	Same pronunciation but different meaning
아갈(slang of mouth)	찾아갈게요, 날아갈듯 … (I will visit you, like flying)	Same pronunciated syllables
호구(pushover)	발보호구(Foot protector)	Same pronunciated syllables
시키(bastard)	시키지마세요(don't make it)	Same pronunciated syllables
니미(slang of Oh,my)	못찾으니미치겠네(I'm sorry I can't find it)	Same pronunciation from wrong punctuation

색골(an oversexed person)	저 보라색골랐는데 (I got that purple one)	Same pronunciation from wrong punctuation
야한(sexy)	도와줘야한다니, 있어야한대요 (I need help, I have to)	Same pronunciation from wrong punctuation
씨방, 시불(shit)	피씨방, 택시불러요 (PC room, call a taxi)	Same pronunciated syllables, Same pronunciation from wrong punctuation
뒤져(go to hell)	마을 뒤져도 안보여요 (I searched the village and couldn't find it)	Same pronunciated syllables
죽어(go to hell)	죽어도 안되겠는데 (Even if I die, it won't happen)	Same pronunciation but different meaning
제길(slang of Oh,my)	언제길러짐 (When am I going to level up?)	Same pronunciation from wrong punctuation

Only a few profanities appear in games, but there are many variations of these. As such, creating a dictionary of prohibited words is inefficient, as shown in **Table 3**. To filter the profanity "개새끼(bastard)" one must go through the trouble of adding all variant profanities to the list of prohibited words. In this manner, a total of 1,959 prohibited words were specified by the game company.

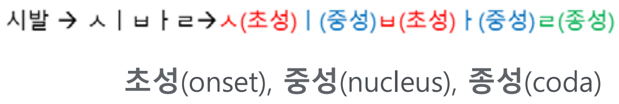
Table 3. Inefficiency in a prohibited word dictionary, especially bastard

Various forms of morphemes for a prohibited word, Bastard		
ㄱ ㅐ	개 ㅅ ㅐ	개넌
ㄱ ㅐ ㅅ ㅐ	개새끼	개넘
ㄱ ㅐ ㅅ ㅐ ㄲ ㅣ	개새끼	개년
ㄱ ㅐ 새	개새끼	개놈
ㄱ ㅐ	개새끼	개놈아
개	개	개뇬
개넘	개 같은	개눔
개년	개같은넘	개늠아
개놈	개같은년	개도라이
개뇬	개같은년	개때까
개늠	개같은놈	개또라이
	개같은뇬	개련

3.2 Features

Unlike in English, some people write Korean words by skipping spaces between words. Therefore, creating tokens corresponding to the spaces may result in problems when building dictionaries, given the style of Korean word endings and postpositional particles. Thus, "나/나는/내가/나를" (I/I am/I am/me) and "먹다/먹는다/먹으니" (eat/eating/eat then) all become different tokens. Therefore, morpheme analyzers are usually employed to separate the basic forms of Korean words, but there are problems with this approach as well because morpheme analyzers are created via feature engineering. These morpheme analyzers cannot incorporate context; therefore, they have limitations in tokenizing words. Moreover, morphemes often suffer from ambiguity. If the morpheme analysis does not fit the context in the preprocessing stage, morphemes with the wrong meanings may be entered into the model. Additionally, it is difficult to properly use spaces in everyday Korean speech. This is because even the standards created by the National Institute of the Korean Language are vague. Therefore, each person performs spacing differently. This causes data to be nonuniform and contributes to the difficulty in automating natural language processing.

Therefore, in this study, we created a dictionary by separating Korean text into graphemes (initial consonant, vowel, final consonant), vectorizing each input value as a number, and using the vectorized number as the model's input. For example, to separate "시발" into graphemes, the process in **Fig. 1** is used.


시발 → ㅅ ㅣ ㅂ ㅏ ㄹ→ㅅ(초성) ㅣ(중성)ㅂ(초성) ㅏ(중성)ㄹ(종성)

초성(onset), 중성(nucleus), 종성(coda)

Fig. 1. Grapheme separation process

In addition to grapheme separation, the characters were separated into syllable units and comparative experiments were performed. For example, in the case of ["시발"], the word was separated into ["ㅅ," "ㅣ," "ㅂ," "ㅏ," "ㄹ"], and it was separated into ["시," "발"]. The input values for these were [6, 5, 12, 3, 8, 0, 0, 0 ...] and [16, 28, 0, 0, 0, 0, 0 ...], respectively. We used 0 as padding based on the longest phrase's input value so that all the phrases would have a fixed length. For the numbers, a dictionary number was assigned according to the frequency of the letter, starting at 0. A dictionary was created to separate the text into grapheme units using information on the Korean initial consonants, vowels, and final consonants, as well as the English letters, numbers, and special characters that appeared in the experiment data, as shown in **Table 4**.

Table 4. Grapheme-separated dictionary example

Dictionary	Frequency	Dictionary No.
'<PAD/>'	1,226,614	0
' '	7,798	1
'ㅇ'	7,716	2
'ㅏ'	6,534	3
'ㄴ'	6,191	4
'ㅣ'	5,001	5
'ㅅ'	3,936	6
'ㅈ'	3,697	7
'ㄹ'	3,506	8
...	...	...
'ㄶ'	1	124
' [UNK] '	0	125

3.3 CNN for Classifying Words

CNNs are mainly used to process image data, but it has been discovered that they also process text data efficiently. Previous studies have reported that CNNs perform better than cutting-edge algorithms in text processing. This is because text can be considered as a one-dimensional image. The reason CNNs are very effective in image processing is that they can recognize patterns. That is, the effectiveness of CNNs stems from the fact that they can find patterns in the locations of pixels such as words, or graphemes rather than only recognize words. For example, they can find patterns among adjacent pixels that are related to other pixels. This is significant because images are not a collection of random pixels; rather, correlated pixels form overall images. To change text into an image form, words must be encoded as numbers. Word embedding is used to depict words as numeric values. Assigning a unique integer value to each word leads to a large number of dimensions, which is termed as one-hot encoding. Therefore, word embedding sets the number of dimensions and uses the values of k dimensions to depict all words. For example, if k = 5, the word "hospital" can be depicted as [0.1, 0.2, 0.3, 0.8, 0]. This embedding method can significantly reduce the size. However, this method has the disadvantage of not considering the relationships between words when arranging them into a matrix. Performance improves further if the relationships between words are considered when the input data is converted into an input matrix like an image.

The deep leaning-based architecture is shown in **Fig. 2**. In this model, we used a CNN after transforming input sentences into a matrix form. Words were converted into k-dimensional vectors, and the number of words to create the input matrix was set as n. The sentences were tokenized, and *n* words were the input values. Because there were differences in length between the sentences, the rows of the last sentence for each sentence were filled with zeros until *n* number of values were filled. This process is called padding.

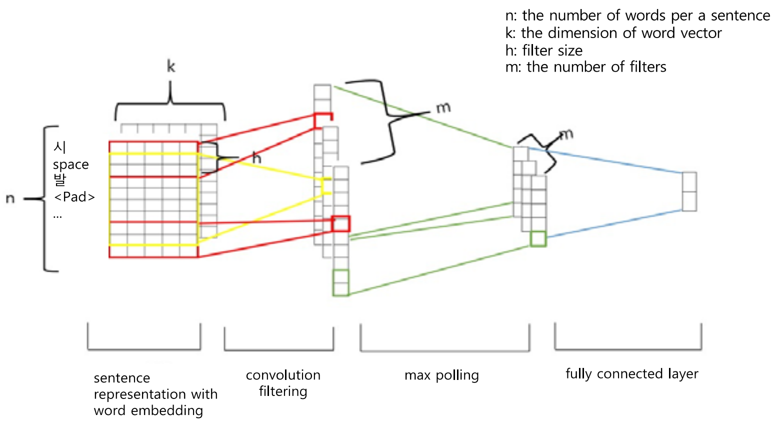


Fig. 2. Deep learning model for text

A hidden layer and a convolutional layer were used in the network to derive abstract features from the input data and automatically derive algorithmically useful features. In the convolutional layer, the input matrix is put into a k × h size filter, and the h × k size input is mapped to the convolutional layer's cells. The square with red lines in **Fig. 3** is the 3 × k filter. The filter operates in units of *h*. As the filter is required to maintain the words' meanings when totaling the words, the filter's horizontal size was set as *k*. Consequently, the convolutional layer is one-dimensional. The number of nodes in a convolutional layer is calculated as n-h+1 because the h-length filter is slided by 1 until *n*. The vertical size of the filter can vary. In **Fig. 2**, the filter sizes are shown in red (size 3) and yellow (size 8). Users can change the number of filters according to their needs. To build a sufficient number of features for a complex problem, more filters are required. The max pooling layer is used to avoid overfitting resulting from using too many features for a given problem. In the max pooling layer, a filter with the size (n-h+1) ×1 is selected, and the maximum values are found for all the values in the preceding convolutional layer. The max pooling layer generates *m* number of one-dimensional rows.

Finally, the m number of rows are fully connected to the output, and this layer is called a classification layer. In the output layer, there are two nodes that represent the gender of the classes. The two nodes are filled with 0 or 1, which represent female or male, respectively

4. Experiments

Instead of image pixels, the input values in most text data tasks are phrases that can be represented by a matrix. Each row in the matrix corresponds to a single token, that is, a word, but it can also be a character. An embedding process is performed to quantify these words as suitable vectors.

We used a 10 × 100 matrix as input for every 10 words within a sentence, which used a 100-dimension embedding. This acts as an image and can be processed by a CNN. In image processing, filters move according to the image's regional filter. However, for text processing, it is common to use a filter that slides over entire rows of the matrix. Therefore, the filter's width is generally the same as the width of the input matrix. The height and size of the regions can vary, but a sliding window of two to five words or more is normal.

We used the network structure of the model that is a convolutional neural network consisting of three convolutional layers and two fully connected layers as shown in **Fig. 3**.

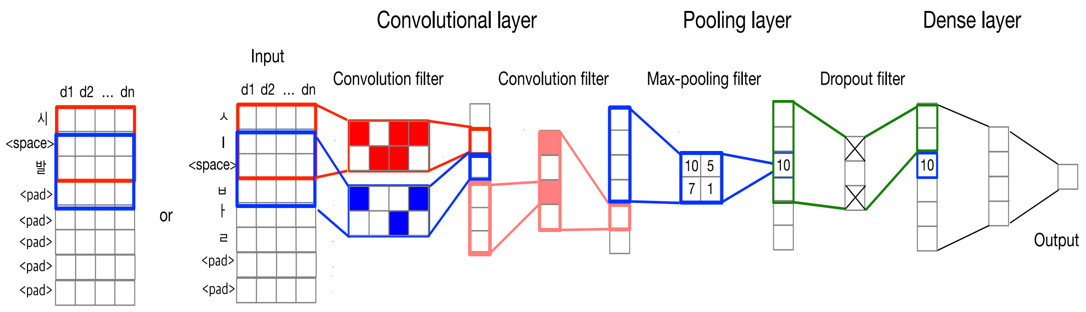


Fig. 3. Proposed deep learning structure for profanity classification

The model input corresponded to the maximum number of syllables when the given chatting data was divided by syllables, and it corresponded to the maximum number of graphemes when the data was divided by graphemes. When the length of the chatting data was short and the number of syllables or graphemes was smaller than the maximum number of words or graphemes, zero was used as padding. Vectors made of zeros were added to text that had small groups of words to match the maximum-word group size, which ensured that the model's input had a fixed size.

The model input was encoded as a 64-dimensional vector and converted to a vector with 64-dimensional properties. It passed through the first and second convolutional layers and was reduced to 16 dimensions. It further passed through the final convolutional layer and was reduced to eight dimensions. At this point, the convolutional layer's filter size was set to 3. In the first and second convolutional layers, the filter moved the vector matrix at random intervals, and simple context information was gathered. The final convolutional layer extracted the profanity word information that influenced classification. The matrix that was produced by the three convolutional layers was sent to the pooling layer. At this point, the model used max pooling, which searched the surrounding values and selected the largest values. The feature vectors that were produced by the pooling layer were reduced to half of their size, and they underwent a flattening process, which converted features to one dimension to send them to the fully connected layer. Softmax was selected as the fully connected layer's activation function, and the model outputted the probability of each sentence being in the normal class or profanity class. Rectified Linear Unit (ReLu) was used as the activation function for all layers except the fully connected layer. An Adam optimizer was used for parameter optimization and L2 regularization was used to avoid overfitting. The recopies of CNN for profanity classification is shown in **Table 5**.

Table 5. Recipes of CNN for profanity classification

Layer(type)	Output Shape	Parameter, #
Embedding_1 (Embedding)	(None, 469, 64)	12800
Convid_1 (conv1D)	(None, 469, 16)	3088
Batch_normalization_1	(None, 469, 16)	64
Convid_2 (conv1D)	(None, 469, 16)	784
Batch_normalization_2	(None, 469, 16)	64
Convid_3 (conv1D)	(None, 469, 8)	392
Max_pooling1d_1	(None, 234, 8)	0
Flatten_1 (Flatten)	(None, 1872)	0
Dense_1 (Dense)	(None, 8)	14984
Dropout_1 (Dropout)	(None, 8)	0
Dense_2 (Dense)	(None, 2)	18
Embedding_1 (Embedding)	(None, 469, 64)	12800

5. Results and discussion

When the phrases from the experimental data were separated into graphemes, 126 graphemes were used in the dictionary. When they were separated into single characters, syllable separation, 1,065 extracted letters were required. In the embedding process, when the phrases were separated into graphemes, a vector length of 469 graphemes was needed for the length of the longest phrase. For syllable separation, a vector length of 241 was needed. Thus, the process was twice as fast for syllable separation than for grapheme separation.

The experiment was conducted after separating the phrases into graphemes and performing embedding. The results are shown in Table 4. When the epoch was 20 and the batch size was 32, the accuracy was 89.04%, and the precision was 90.93% for profanity and 83.39% for normal class. The recall was 94.25% for profanity and 75.4% for normal class. The F1 score was 92.56% for profanity and 79.04% for normal words. The performance of profane class outperforms the normal class.

As shown in **Table 6**, the frequency of spaces in the created dictionary was high. The results of removing the space features and performing experiments are shown in **Table 7**. The accuracy reduced, albeit slightly, and it can be seen that spaces are an important feature for judging profanity.

Table 6. Grapheme separation results

	Profane class	**Normal class**
Precision	90.93%	83.39%
Recall	94.25%	75.43%
F1 measure	92.56%	79.21%
Accuracy	89.04%	

Table 7. Space removal results

	Profane class	**Normal class**
Precision	89.56%	81.79%
Recall	93.92%	71.39%
F1 measure	87.68%	87.68%
Accuracy	87.68%	

Next, we modified the structure of the CNN to improve the performance. Although it is only a slight improvement compared to the results in **Table 8**, performance was improved by approximately 1%. It was found that global max pooling with a single convolutional layer produces the best performance for this data.

Table 8. Grapheme separation CNN structure results

CNN structure	**Accuracy**
Conv+Conv+Pool+Dropout+FC	86.88%
Conv+Pool+Dropout+FC	85.6%
Conv+Conv+Conv+Pool+FC+Dropout+FC	89.04%
Conv+globalpool+FC+Dropout+FC	**90.4%**
Conv+Conv+globalpool+FC+Dropout+FC	90.08%
Conv+Pool+Conv+Pool+FC	86.4%
Conv+Pool+Conv+Pool+Conv+Pool+FC 85.7%	85.7%

The results of the experiment that separated the text into syllables are shown in **Table 9**. Similar to the experiment that separated the text into graphemes, the epoch was 20, and the batch size was set to 32 with the best model of **Table 8**. In this experiment, the accuracy was 89.36%, and the precision was 92.22% for profanity and 81.60% for normal words. The recall was 93.14% for profanity and 79.48% for normal words. The F1 scores were 92.68% for profanity and 80.53% for normal words.

Table 9. Syllable separation results

	Profane class	**Normal class**
Precision	92.22%	81.60%
Recall	93.14%	79.48%
F1 measure	92.68%	80.53%
Accuracy	89.36%	

The experiments show that separating text into syllables is little more accurate than grapheme separation in recognizing normal words from filtered words as profane using the list of profanity words. Through these experiments, we found that syllable or grapheme separation word embedding performs well the Korean language that can exploit morpheme-based profane detection.

6. Conclusions

In this paper, we proposed a method of discerning profanity using Korean-language game chat data. Our contribution is that we tickled that the dictionary-based profane detection widely used in the industry has a weakness because of morpheme-diversity of Korean spoken language. Dictionary filters can detect normal text as profanity due to the coincidence of letters. To resolve the problem, we suggested the use of grapheme separation and syllable separation to create a dictionary for profanity analysis, as well as using the traditional morpheme-based separation. Using that, we provided a way to improve quality to the filtering algorithm currently adopted in most online services. Experiments on chat dialogs in a game show that the accuracy of the method was improved slightly by not separating the text into graphemes, but the number of dictionary entries was reduced by more than a factor of 10 when the text was separated into graphemes. Furthermore, separating text into graphemes produced better results in terms of normal text classification performance.

Follow-up studies will be required to compare the results of experiments that remove special characters other than spaces.

Second, previous studies have found that when Korean phrases are classified using both word and grapheme information, classification performance is higher than conventional models that use only word information or grapheme information [19]. In this study, it also appears that a higher performance can be obtained by combining grapheme-and syllable-based techniques. Additionally, if a word-based profanity dictionary is created, further research can be performed based on related studies.

Third, more advanced deep learning techniques such as BERT to investigate whether advanced model can recognize the difference between normal words and profane words that are literally the same without grapheme-level separation.

Acknowledgement

This study is supported by Korea University Grant and in part by the Soonchunhyang University Research Fund 20180051.

References

[1] S.M Park, "2019 Cyber Report ③ A pen is stronger than a sword, a keyboard is more dangerous than a gun," *SISAWEEK*, 68, Kyonggidae-ro, Seodaemun-gu, Seoul, Republic of Korea, 2019.

[2] Y.R. Cho, "[NDC2018] Profanity detector that catches s-111 shots, making it with deep learning," 25, Pangyo-ro 256beon-gil, Bundang-gu, Seongnam-si, Gyeonggi-do, Republic of Korea, 2018.

[3] S. Sood, J. Antin, E. Churchill, "Profanity use in online communities," in *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pp.1481-1490, 2012. Article (CrossRef Link)

[4] Y. Chen,Y. Zhou, S. Zhu, H. Xu, "Detecting offensive language in social media to protect adolescent online safety," in *Proc. of the 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pp.71-80, 2012. Article (CrossRef Link)

[5] C. Nobata,J. Tetreault, A. Thomas, Y. Mehdad, Y. Chang, "Abusive language detection in online user content," in *Proc. of the 25th international conference on world wide web*, pp.145-153, 2016. Article (CrossRef Link)

[6] K. Park, J. Lee, “Developing a Vulgarity Filtering System for Online Games using SVM," in *Proc. of the Korean Institute of Information Scientists and Engineers conference*, vol. 33, no. 2, pp. 260-263, 2006.

[7] H.-S. Lee, H.-R. Lee, J.-U. Park, Y.-S. Han, "An abusive text detection system based on enhanced abusive and non-abusive word lists," *Decision Support Systems*, vol. 113, pp. 22-31, 2018. Article (CrossRef Link)

[8] M. Märtens, S. Shen, A. Iosup, F. Kuipers, "Toxicity detection in multiplayer online games," in *Proc. of the 2015 International Workshop on NetGames*, pp. 1-6, 2015. Article (CrossRef Link)

[9] P. Ratadiya, D. Mishra, "An Attention Ensemble Based Approach for Multilabel Profanity Detection," in *Proc. of the 2019 ICDMW*, pp.544-550, 2019. Article (CrossRef Link)

[10] A.M. Founta, D. Chatzakou, N. Kourtellis, J. Blackburn, A. Vakali, I. Leontiadis, "A unified deep learning architecture for abuse detection," in *Proc. of the 10th ACM Conference on Web Science*, pp. 105-114, 2019. Aricle (CrossRef Link)

[11] H. Yenala, A. Jhanwar, M.K. Chinnakotla, J. Goyal, "Deep learning for detecting inappropriate content in text," *International Journal of Data Science and Analytics*, vol. 6, no. 4, pp. 273-286, 2018. Article (CrossRef Link)

[12] H.-P. Su, Z.-J. Huang, H.-T. Chang, C.-J. Lin, "Rephrasing profanity in chinese text," in *Proc. of the First Workshop on Abusive Language Online*, pp.18-24, 2017. Article (CrossRef Link)

[13] T.-J. Yoon, H.-G. Cho, "The Online Game Coined Profanity Filtering System by using Semi-Global Alignment," *The Journal of the Korea Contents Association*, vol. 9, no. 12, pp. 113-120, 2009. Aritcle (CrossRef Link)

[14] S.Y. Kim, J.Y. Lee, "Fact finding surveying adolescents's language and culture in online games and a countermeasure strategy," *The Journal of Korean Association of Computer Education*, vol. 16, no. 1, pp. 33-42, 2013.

[15] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[16] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, N. Bhamidipati, "Hate speech detection with comment embeddings," in *Proc. of the 24th international conference on world wide web*, pp. 29-30, 2015. Article (CrossRef Link)

[17] S. Seo, S.-B. Cho, "A Transfer Learning Method for Solving Imbalance Data of Abusive Sentence Classification," *Journal of KIISE*, vol. 44, no. 12, pp. 1275-1281, 2017. Article (CrossRef Link)

[18] T. Chu, K. Jue, M. Wang, "Comment abuse classification with deep learning," Stanford University, 450 Serra Mall, Stanford, CA 94305, United States, 2016.

[19] Y. Kim, D. Ra, S. Lim, "Zero-anaphora resolution in Korean based on deep language representation model: BERT," *ETRI Journal*, vol. 43, no. 2, pp. 299-312, 2021. Article (CrossRef Link)

[20] M. Song, H. Park, K.-s. Shin, "Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in Korean," *Information Processing & Management*, vol. 56, no. 3, pp. 637-653, 2019. Article (CrossRef Link)

[21] Q.-P. Nguyen, A.-D. Vo, J.-C. Shin, C.-Y. Ock, "Effect of word sense disambiguation on neural machine translation: A case study in Korean," *IEEE Access*, vol. 6, pp. 38512-38523, 2018. Article (CrossRef Link)

[25] J.J. Lee, S.B. Kwon, S.M. Ahn, "Sentiment Analysis Using Deep Learning Model based on Phoneme-level Korean," *Journal of Information Technology Services*, vol. 17, no. 1, pp. 79-89, 2018. Article (CrossRef Link)

Jiyoung Woo received the B.S., M.S., Ph.D degree in industrial engineering from KAIST in 2000, 2002, and 2006. She is a currently professor at Big Data Engineering department, Soonchunhyang University. From 2008 to 2010, she was a researcher with AI lab of Arizona University, USA. She was a research professor at Graduate School of Information Security, Korea University from 2010 to 2016. Her research interest includes game log and medical data analytics.

Sung Hee Park received the B.S. degree in mathematics, M.S. degree in Big Data Engineering from Soonchunhyang University in 2016 and 2018. She is a currently data analyst at Hanwha Investment & Securities Co. Her research interest includes social media and stock data analytics.

Huy Kang Kim received his B.S. degree in Industrial Management in 1998, M.S. and Ph.D degrees in industrial and systems engineering from KAIST in 2000 and 2009. He founded A3 Security Consulting, the first information security consulting company in Korea in 1999. Currently he is a professor in School of Cybersecurity, Korea University.
Before joining Korea University, he was a technical director (TD) and a head of information security department of NCSOFT (2004~2010). His research interests include solving security problems in online games based on the user behavior analysis and data mining.