

## Korean Malicious Text Analysis Technique Applying Consonant and Vowel Separation Method

SungRyeol Yang<sup>1</sup>, Bokyoon Na<sup>2\*</sup>

### Abstract

*Social costs due to malicious writing, including chatting and commenting, are increasing. There are many techniques to filtering malicious writing, but there are problems that make it inefficient because of detection bypass techniques such as typos and misspellings, which make detection avoidable. Since these sentences cause problems even in the preprocessing process through morpheme analysis a preprocessing technique was proposed that applies a particle that separates consonants and vowels. Among the various methods in this paper, the first consonant, middle consonant, final consonant, and non-complete Hangeul consonant form that do not include Korean spaces showed the best performance with an accuracy of about 78%.*

**Keywords:** Korean malicious text analysis, separates consonants and vowels method, 1-D CNN, embedding, preprocessing.

### 1. Introduction

Natural language processing is a field of artificial intelligence that studies and implements human language phenomena so that computers can understand them. Human language is composed of a combination of words, grammar, and spelling, but in actual communication, grammar and spelling may not match. Even if grammar and spelling do not match, there is no big problem in communication. Korean is known to be difficult for analysis due to the following reasons. Korean is an agglutinative language whose meaning changes depending on the addition of suffixes (e.g., “포도” in Korean is “grape” in English. 포도<sub>를</sub>[Podoreul](object), 포도<sub>가</sub>[Podoga](subject)), and is understandable even when the word order changes(e.g., I read a book about natural language processing. In Korean, 나는 자연어처리에 대한 책을 읽는다 → 자연어처리에 대한 책을 나는 읽는다 is same meaning.). Also, Korean have ambiguous spacing rules (e.g., Example: Korean Spelling Rules Chapter 5, Section 4, Paragraph 50. In principle, technical terms should be written in spaces for each word, but they can be used without spaces [1].), no difference between declarative and interrogative sentences, and it can be meaningful even without a subject (e.g., 책 읽었어. (I read a book.), 책 읽었어? (Did you read a book?). In addition to these characteristics, it is used to circumvent the Korean malicious text detection technique by transforming it into a similar character (e.g., 시 → ㅅㅣ) or replacing it with a similar pronunciation (e.g., “안녕” in Korean is “Hello” in English.

<sup>1</sup> Department of Computer Engineering, Tech University of Korea, Republic of Korea, troubleshooter@tukorea.ac.kr

<sup>2</sup> Department of Computer Engineering, Tech University of Korea, Republic of Korea, bkna@tukorea.ac.kr

안녕[An Nyeong] → 안뇽[An Nyong]). Korean language is difficult to analyze to begin with, which makes analyzing malicious writings in Korean even harder.

The social cost of malicious posts can not be ignored. According to Yonsei University's Barun ICT Research Center, the social and economic costs due to malicious comments reach up to 35 trillion KRW (about 26 billion USD) annually [2]. In addition to malicious comments, there are many other routes. According to the "2021 Cyber Violence Survey Report [3]" provided by the Korea Communications Commission, text and instant messages appear to be the highest route of violence, followed by SNS and online games, so there is a steady demand for analysis of malicious texts.

The general sequence of natural language processing is as follows. First, it goes through the preprocessing process, then the tokenization process, then the embedding process, and then learns. Morphological analysis is mainly used in the tokenization process for Korean text analysis, but due to the nature of online chatting or comments, it does not show good performance to use malicious chat detection bypass techniques such as typos. Therefore, a new technique is needed, such as the consonant and vowel separation method, rather than the existing method. [4].

***Representative existing methods for analyzing malicious texts are as follows.***

1. Prohibited Word-Based Filtering System

Among the techniques for blocking malicious text, the most common method is a prohibited word-based detection rule. A prohibited word-based detection rule simply designates a specific word as a prohibited word and considers it to be malicious if the word is included. However, this technique has clear limitations. For example, in Korean, there is a “시발역” which means the station from which a train departs. This word is also listed in the standard Korean dictionary, but it is caught because it contains the slang word ‘시발(meaning bad in English)’ in the prohibited word-based filtering. In this case, it can be solved by exception handling, but in actual cases, it is very difficult to solve with simple exception handling. As another example, if “쓰레기(trash)” is a forbidden word, exceptions should be made for various words such as “쓰레기장(garbage dump)” and “쓰레기통(trash can)”, and typos such as “쓰래기(shapes are similar)” should be considered. In addition, when people use “시래기[si-rae-gi], a word with a similar pronunciation to “쓰래기[ssu-rae-gi], as a slang word, it has a limit that is difficult to handle. Increasing the detection sensitivity increases the false positive rate, and it is an easy system to circumvent detection.

2. Korean Morphological Analysis

There have been many attempts to analyze Korean morphemes in the past. Representatively, there is the KoNLPy package [5], which is a Korean morpheme analyzer in Python. The KoNLPy package is a Python package that performs morphological analysis and POS tagging. Morphological analysis means identifying the structure of various linguistic attributes, such as morphemes, roots, prefixes, suffixes, and parts of speech. POS tagging requires a proper understanding of Korean grammar, and the speed is greatly affected by the analysis method and quality [Figure 1]. Here, problems arise during this analysis due to the characteristics of malicious writing in Korean (using detour techniques such as initial consonants, grammar, and misspellings).

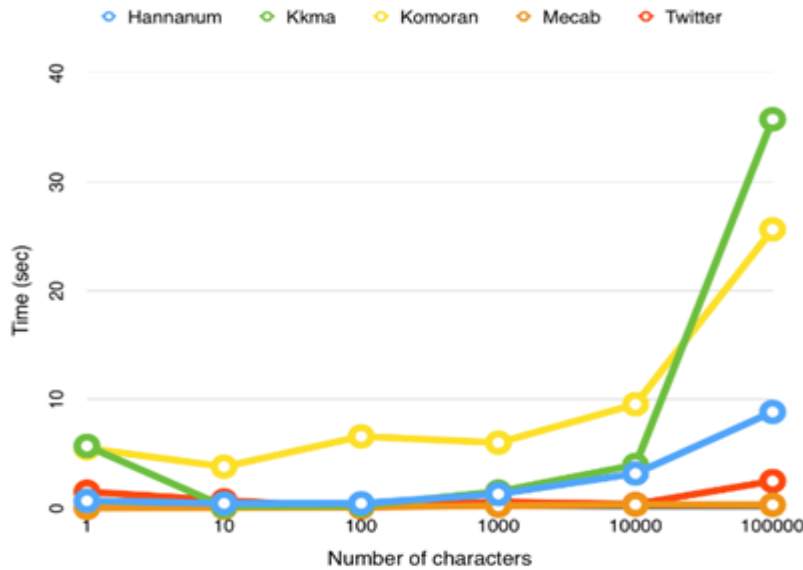


Figure 1: Speed per package in KoNLPy provided by KoNLPy site [6]. As the number of characters increases, the execution time usually increases rapidly.

First, there is a problem in that the execution time increases rapidly as the text becomes longer. In addition, if you use a relatively slow but somewhat accurate Kind Korean Morpheme Analyzer(Kkma) method when analyzing Korean morphemes, you can see that `java.lang.OutOfMemory` appears for malicious articles that are simply repeated without spaces such as “ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ (ㄷ ㄷ means tremble in English)”. This is a known bug [7] but difficult to solve. Therefore, depending on the method of the morpheme analyzer, a weak problem may exist during morpheme analysis.

### 3. Alphabet Separation Techniques

Semantic analysis, a technique that analyzes the context and identifies it, seems to be a very suitable technique for determining whether it is malicious, but it is not widely used due to difficulties in computing power, data set construction, and modeling, and is a technique in progress. Therefore, it was not considered in this study.

### 4. Alphabet Separation Techniques

It is simply a method of separating letters. For example, it is separated into (glass  $\rightarrow$  g l a s s), and in the case of Hangeul, in which vowels and consonants are combined to express one character, it is separated into (“유리” means glass in Korean. 유리  $\rightarrow$  ㅇ ㅠ ㄹ ㅣ ). According to the lecture at NDC2018 [4], which is the basis of the idea of this paper. It shows that the analysis of malicious writing in Korean is effective by using techniques such as the Alphabet Separation Techniques method and Attention Mechanism together.

Although there are many attempts to analyze malicious Korean writings as above, there are many cases where the analysis is not properly performed due to technical limitations or the characteristics of Korean writing. There is also a problem that morphological analysis takes a long time even if it is performed. To solve the problem that morpheme analysis does not work properly and the problem that it takes a long time, as a method to replace morpheme analysis, the method of separating consonants and vowels was adopted.

In this study, an improved method of separating consonants and vowels was proposed to replace the difficulty of morpheme analysis in the tokenization process of Korean natural language processing analysis. This method is done by attaching a symbol that can distinguish a single letter such as “ㅠ”, “ㄱ” or “2”, and a last consonant letter in addition

to the simple consonant and vowel separation method. This is to show better accuracy than simple consonant and vowel separation methods even when using malicious chat detection bypass techniques (such as typos, initial consonants, and similar-shaped special characters), which are characteristics of malicious articles, and to enable analysis with simple CNN models. It is basically inferred that the use of simple CNN models also uses less computing power for learning than complex models. For reference, the Korean text of this study mainly refers to short but weak grammatical elements such as Korean chat and comments.

## 2. Contents

### 2.1 Proposed Method

The proposed method enables the separation of consonants and attachment of separators so that they can distinguish between last consonant letter and non-complete characters. The previously referenced [4] method of separating consonants was introduced only by simply separating consonants. Considering that there will be performance changes depending on the consonant and vowel separation method, the researchers tried various modified consonant and vowel separation methods. The common consonant and vowel separation method used in the experiment usually separates consonants, including all letters of a given sentence, rather than tagging them through morpheme analysis by preprocessing that excludes some characters such as special characters. The reason for not excluding some characters in the sentence is as follows. Due to the nature of malicious writing in Korean, there are many cases in which special characters are expressed as part of a letter (e.g., the Korean letter “시” is expressed as “ㄱ!”, “ㄱ 1”, “^ | ”, etc.). In addition, even if it is a simple repetitive letter, it should not be excluded. It has more stronger meaning than general sentences, so excluded repetitive letter may cause reduce meaning (e.g., the sense of tone changes according to the number of ㄷ [8]). Also, since spacing was introduced in Korean by a foreign missionary in 1877 [9]. It was excluded because the importance of spacing was relatively low compared to other Korean grammars. To measure the performance of the wave method to separate consonants, the experiment was conducted in the following way. The proposed method corresponds to number 5 of the following experimental methods.

1. A simple consonant and vowel separation method including spacing in Korean sentence (e.g., meaning “starting during break time” in Korean online game chat, 휴식 중 ㄱㄱ → ㅎㅍㅅ | ㄱ / ㅈㅊㅇ / ㄱㄱ) (“ / ” means space in separated example, not actually output. It is a code entered because it is not possible to identify spaces.)
2. A simple consonant and vowel separation method that does not include spaces in Korean sentence (e.g., 휴식 중 ㄱㄱ → ㅎㅍㅅ | ㄱㅈㅊㅇㄱㄱ)
3. A consonant and vowel separation method that does not include Korean spaces and matches the position of Korean initial consonants, vowel, and final consonants (e.g., 휴식 중 ㄱㄱ → ㅎㅍ / ㅅ | ㄱㅈㅊㅇ ㄱㄱㄱㄱㄱ). (“ / ” means space in separated example, not actually output. It is a code entered because it is not possible to identify spaces.) Since the initial consonant, vowel, and final consonant is repeated every 3 tokens, people can immediately know which token is an initial consonant, a neutral consonant, or a final consonant by looking at the data type. However, if it is not a complete combination of Korean consonants and vowels, insert tokens at all positions of initial consonants and final consonants. This is because we thought it had a strong meaning like an emoticon unless it was a special character or a complete letter.

4. A consonant and vowel separation method excluding final consonants in Method 3. When communicating in Korean, it is possible to communicate without difficulty even if the final consonants are incorrect (e.g., meaning “Hello, nice to meet you” in Korean, please pay attention to the similarity of the pronunciation in [ ]). Correct sentence is 안녕하세요 만나서 반가워요 [annyeonghaseyo mannaseo bangawoyo], and incorrect final consonants like안녕하섯용 만나섯 방가웁요[annyeohhas-sess-yong mannaseos bang-gawong-yo] or 아녀하세요 마나서 바가웁요[anyehaseyo manaseo bagawoyo]). However, if the initial consonant and the vowel consonant change, it becomes very difficult to understand the meaning (e.g., meaning “Hello” in Korean, please note that the pronunciation in [ ] is very different from the original sentence. Correct sentence: 안녕하세요[ann-yeong-ha-se-yo], incorrect initial consonant and vowel sentence: 언녕하셔여[eon-nyang-ha-syeo-yeo]), so we experimented with excluding the final consonant. In other words, since it was thought that initial consonants and vowels had the most influence in Korean communication, only final consonants, which were considered relatively unimportant, were excluded.

5. A form that does not include spaces in Korean sentences and distinguishes initial consonants, vowels, final consonants, and those that are not in the complete Korean consonant and vowel form. (e.g., meaning “Hi 😊” in Korean, sometimes used in Korean online chat. 하2용ㅎ → ㅎ[2]ㅇㅍ{ㅇ}[ㅎ])

6. A form that includes spaces in Korean sentences and distinguishes initial consonants, vowels, final consonants, and those that are not in the complete Korean consonant and vowel form. (e.g., meaning “Hi 😊” in Korean, sometimes used in Korean online chat. 하2용ㅎ → ㅎ[2]ㅇㅍ{ㅇ}[ㅎ])

Like method 3, but the case including spaces was intentionally not considered. The number of “ ” tokens increase sentence’s length too much, and it is not clear where the initial consonants, vowels, and final consonants are located due to spacing. Method 5 and 6, for example, “[ㅎ]” were processed as “[”, “ㅎ”, and “]” instead of a group. These methods distinguish not only initial consonant sounds, vowels, or final consonants, but also those that are not in the complete Korean consonant and consonant form.

## 2.2 Used Dataset

The dataset used was the text ethics verification data [10] provided by AI Hub. Since the text ethics verification data [10] provides a train dataset and a test dataset, respectively, data that avoids duplication can be used for train and test. Some of this data is mixed with initial consonants (e.g., This example sentence contains swear words in Korean, so we will not display the meaning in English. Example sentence: 씨ㄴ 뭐래), or data mixed with non-Hangul symbols (e.g., This example sentence contains swear words in Korean, so we will not display the meaning in English. Example sentence: 엠2). There is also a characteristic that many misspelled data exist. It is difficult to judge whether the writing is malicious in Korean because each person has different standards. This dataset [10] used a voting system to solve this problem and provided voter information together. However, to prevent social controversy, certain names were converted to “#@인간및인간집단.인물.유명인#” (in English, “#@human and human group.person.celebrity#”) and provided name information before being converted to the safe zone. However, in this study, the researchers did not link the information to the safe zone.

The label data distribution structure of the used dataset is shown in Figure 2.

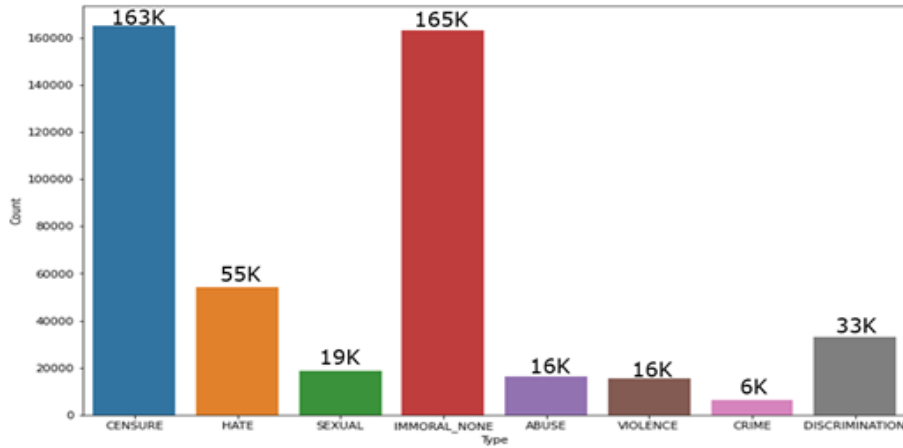


Figure 2: This is the result of analyzing the train dataset. Only up to 1000 units were indicated.

From this data, it is only necessary to determine whether it is malicious or not. If there is only IMMORAL\_NONE, it is reclassified as 0 because it is a normal sentence. If even one of the remaining categories is included, it is reclassified as 1 because it is a malicious sentence. After removing duplicate data, the data were cut to a smaller number so that the ratio of data was 1:1, as shown in Table 1 and Table 2, to solve the data imbalance (use Under Sampling technique).

Table 1: Label distribution of modified train data sets

label (0: normal, 1: malicious)	Count (sum: 325201)	Under Sampling (sum: 284614)
0	142307	142307
1	182894	142307

Table 2: Label distribution of modified test data sets

label (0: normal, 1: malicious)	Count (sum: 68147)	Under Sampling (sum: 57052)
0	28526	28526
1	39621	28526

### 2.3 Applying Preprocessing

After the process of adjusting the ratio of the data, the consonants of the sentence were separated and stored for preprocessing. When the consonants of the learning data were separated and preprocessed to be used for learning, the results of the data were divided as shown in Table3 and Table 4 for each preprocessing technique.

Table 3: Results of performance comparison by preprocessing method using modified train data set

Category	Method 1	Method 2	Method 3	Method 4
Maximum length	1245	1119	1285	935
Average length (rounded)	97.4002	86.6887	101.7233	74.0646
Average length of normal sentence (rounded)	85.3869	76.1042	89.7031	65.3229
Average length of malicious sentence (rounded)	109.4135	97.2731	113.7436	82.8063
The vocabulary size in the train data	118	118	118	109
The number of rare words that occur less than once	14	14	1	0
Percentage of rare words in vocabulary (rounded up)	11.8644	11.8644	0.8475	0.0000
Percentage of rare word appearances in total appearance (rounded)	0.0001	0.0001	8.0300	0.0000

Vocabulary size (Calculated method)	120(118+2)	120 (118+2)	120(118+2)	111(109+2)
Shape of training data (padded shape)	(284614,200)	(284614,200)	(284614,200)	(284614,200)

\*Note: The threshold for deciding rare words was set to 2. "Method 1" to "Method 4" means the experimental methods 1 to 4 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 4" due to insufficient table space.

Table 4: Label distribution of modified test data sets

Category	Method 5	Method 6
Maximum length	1351	1477
Average length (rounded)	104.4738	115.1854
Average length of normal sentence (rounded)	91.7031	110.9857
Average length of malicious sentence (rounded)	117.2446	129.3850
The vocabulary size in the train data	118	118
The number of rare words that occur less than once	14	14
Percentage of rare words in vocabulary (rounded up)	11.8644	11.8644
Percentage of rare word appearances in total appearance (rounded)	0.0001	0.0001
Vocabulary size (Calculated method)	120(118+2)	120(118+2)
Shape of training data (padded shape)	(284614,200)	(284614,200)

\*Note: The threshold for deciding rare words was set to 2. "Method 5" to "Method 6" means the experimental methods 5 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 5" to "Method 6" due to insufficient table space.

The time required for preprocessing the consonant and vowel separation method using the entire train dataset is less than 20 seconds Figure 3 based on the free version of Google Colab [11]. This preprocessing Methods is much faster than Morphological analysis using KoNLPy Figure 1.

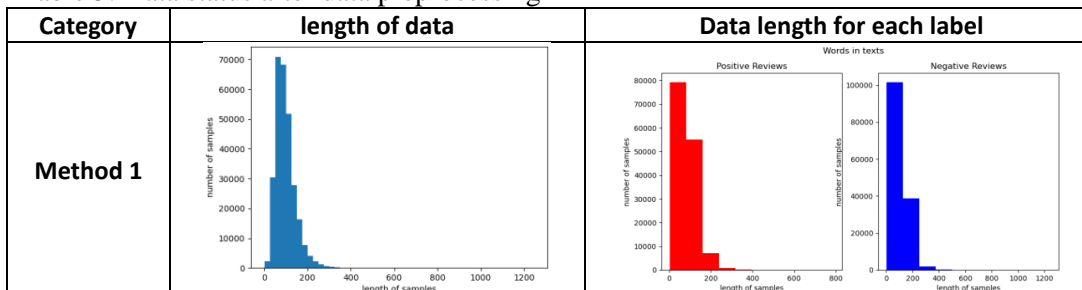
```

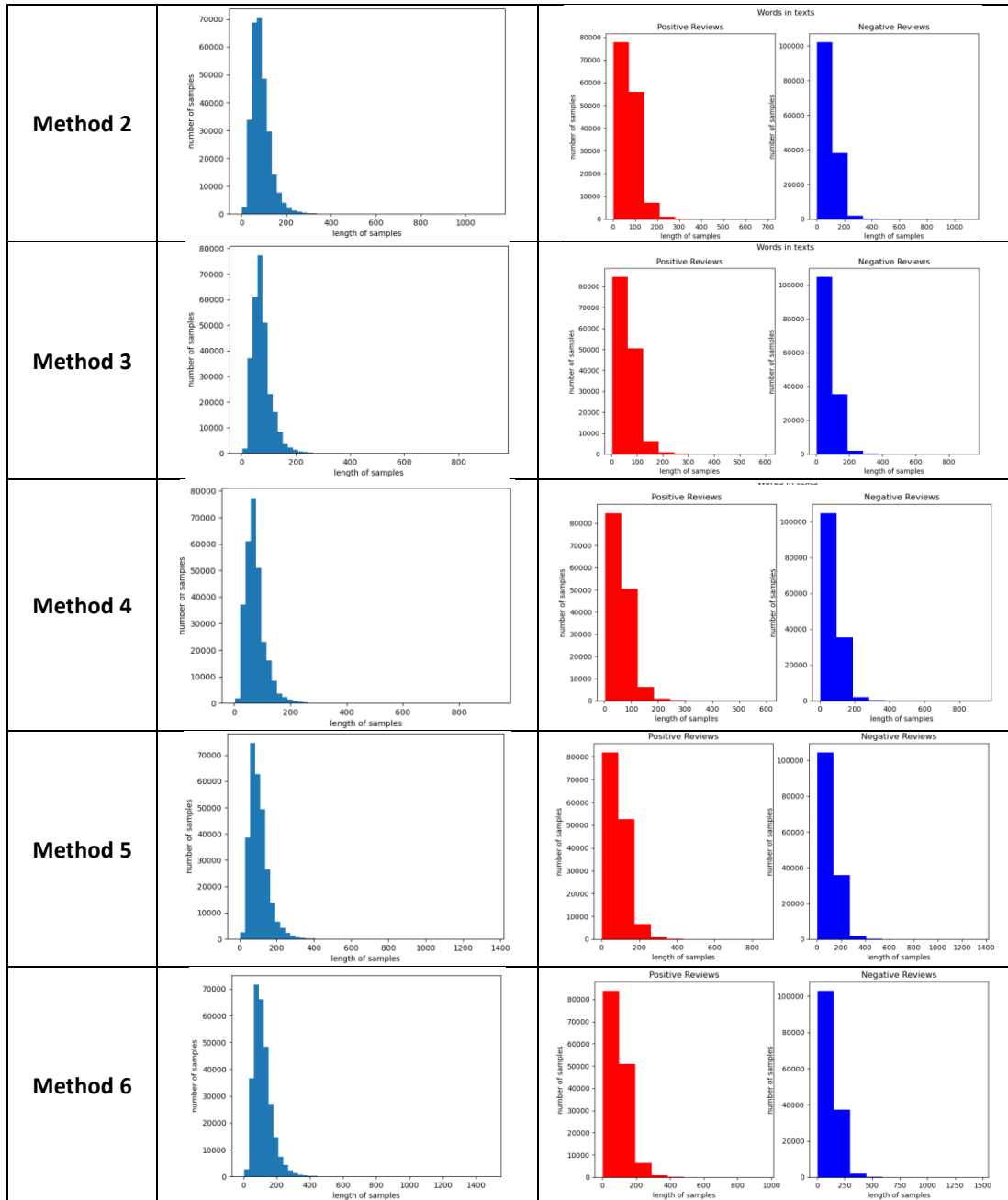
1 #한국어 띄어쓰기 포함
2 start = time.time() #시작 시간 저장
3
4 data["Preprocessed"] = data["content"].apply(lambda x: ' '.join(s for s in convert(x)))
5
6 print("WorkingTime: {} sec".format(time.time() - start)) # 현재시각 - 시작시간 = 실행 시간
WorkingTime: 19.438249349594116 sec
    
```

Figure 3: The longest conversion time when each of the proposed preprocessing methods was tried 5 times.

Data status after data preprocessing is shown in Table 5 and Table 6 below.

Table 5: Data status after data preprocessing





\*Note: The six methods in the category are the same as those in the 2.1 proposed method.

### 2.4 CNN Structure Used for Training

For comparison of the methods proposed in 2.1, the structure of the 1-D CNN used for training was fixed as follows [Figure 4].



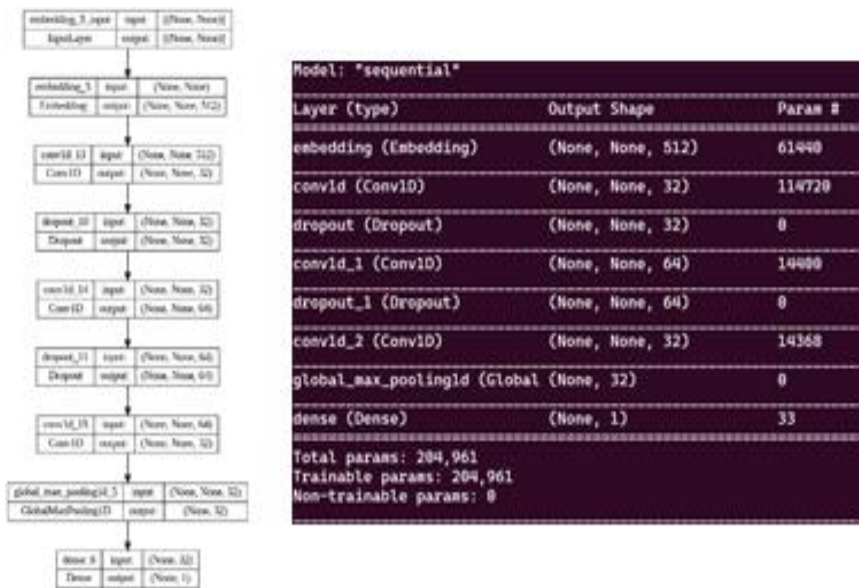


Figure 4: CNN model structure via "tensorflow.keras.utils.plot\_model()" and "tensorflow.keras.models.Sequential().summary". For the reason why the embedding dimension was set to 512, refer to stackoverflow [12].

### 2.5 Hyperparameters

Based on the CNN structure described in 2.3, the remaining hyperparameters which are not introduced yet are as follows. The embedding dimension was set to 512, and the validation split was set to 0.3 in the train data for validation. Since the dropout technique [13] was used, the dropout ratio was set to 0.2. The batch size was set to 64, which is the second largest among the generally recommended 16, 32, 64, and 128, for fast learning and generalization. Since the consonants are separated, we set the Kernel size was set to 7 because the need to perform a convolution operation on several values was considered. Since the consonants and vowels are separated, the order seems to be important, so the padding in Conv1D [14] is set to 'causal'. There was not a big difference in performance between 'valid' and 'causal', which are the padding argument values of keras.layers.Conv1D [14], but in this experiment, 'causal' generally showed better performance than 'valid'. For early stopping, monitor was set to val\_loss and patience to 10. The optimizer used Adam, and the learning\_rate used the keras default value of 0.001 [15]. The reason why the data padding length is set to 200 characters is that most of the data after preprocessing is within 200 characters. In addition, based on data researched on Twitter [16], it was concluded that there are almost no sentences with more than 140 characters in Korean online. Based on these two grounds, it was padded with 200 characters in consideration of the consonant and vowel separation method.

### 3. Conclusions

This experiment was conducted with a total of six methods using the wave method. The experiment was performed 18 times in total, 3 times each with 6 methods. Test data was also used after conversion in the same way as the preprocessing method used for train data. Among the preprocessing methods performed for performance comparison, method 5, "A form that does not include spaces in Korean sentences and distinguishes initial consonants, vowels, final consonants, and those that are not in the complete Korean consonant and vowel form" had an accuracy of about 0.78 for all three learnings. It was stable and showed the highest performance.

The results of the entire experiment were divided into different parts as shown in Table 6 to Table 14, and Figure 5 to Figure 7.

Table 6: Experimental results for each pre-processing process of the 1st experiment

Category	Method 1			Method 2			Method 3		
Epoch	47			54			52		
Accuracy	0.778150			0.77234			0.754750		
Precision	0.836678			0.855219			0.875090		
Recall	0.691229			0.656629			0.594335		
F1-score	0.757031			0.742881			0.707891		
Cohens kappa	0.556300			0.545467			0.509500		
ROC AUC	0.865448			0.868518			0.866195		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	24677	3849	Positive	25355	3171	Positive	26106	2420
	Negative	8808	19718	Negative	9795	18731	Negative	11572	16954

\*Note: "Method 1" to "Method 3" means the experimental methods 1 to 3 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 3" due to insufficient table space.

Table 7: Experimental results for each pre-processing process of the 1st experiment

Category	Method 4			Method 5			Method 6		
Epoch	36			50			59		
Accuracy	0.765144			0.788754			0.737958		
Precision	0.816266			0.812813			0.893279		
Recall	0.684323			0.750298			0.540489		
F1-score	0.744494			0.780306			0.67381		
Cohens kappa	0.530288			0.577508			0.475917		
ROC AUC	0.848787			0.868742			0.865951		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	24132	4394	Positive	23597	4929	Positive	26684	1842
	Negative	9005	19521	Negative	7123	21403	Negative	13108	15418

\*Note: "Method 4" to "Method 6" means the experimental methods 4 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 4" to "Method 6" due to insufficient table space.

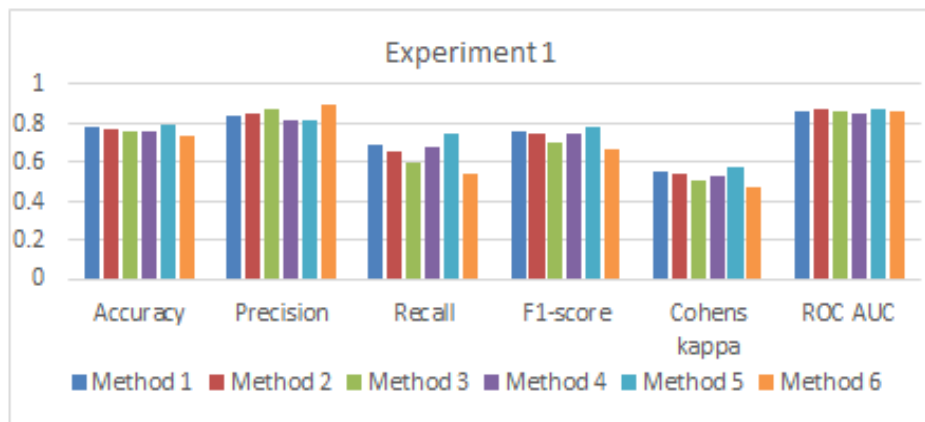
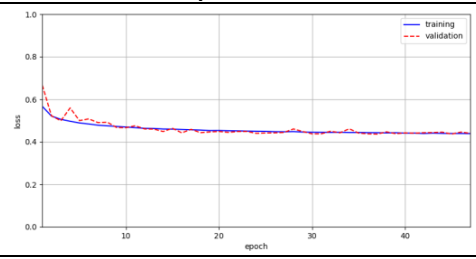
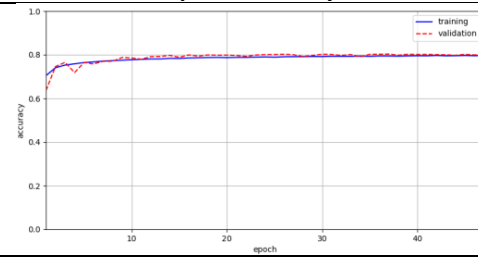
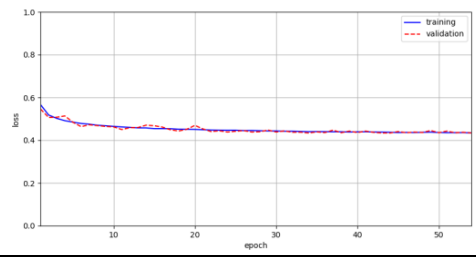
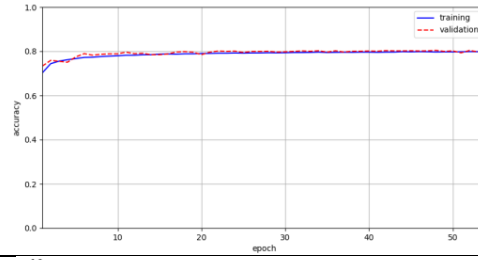
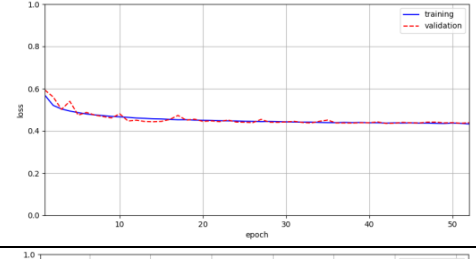
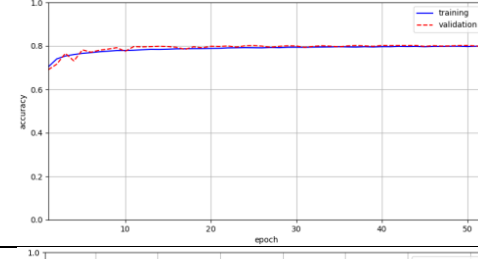
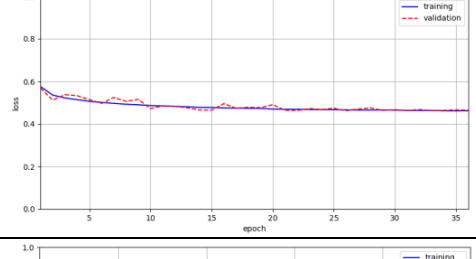
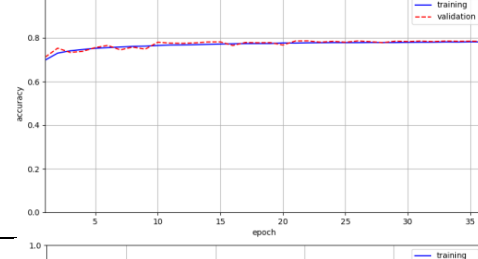
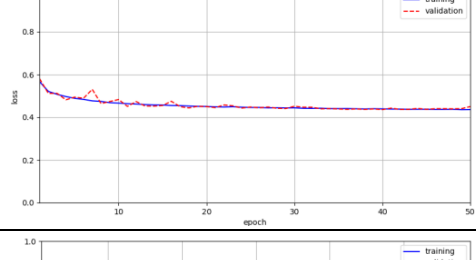
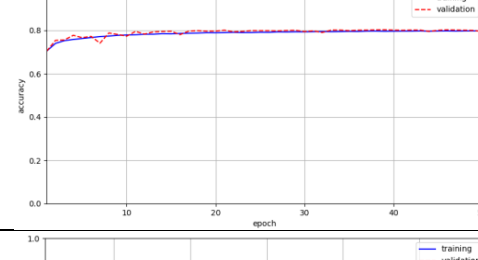
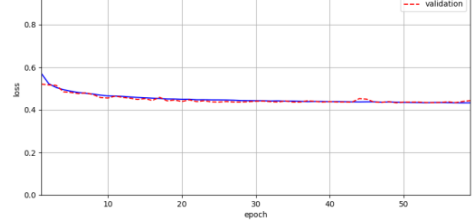
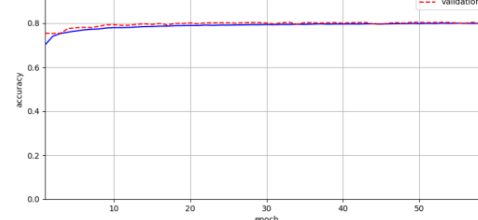


Figure 5: Comparison of all results from Experiment 1

Table 8: Visualization of the learning process for each preprocessing process of the 1st experiment

	Epoch-loss	Epoch-accuracy
<b>Method 1</b>		
<b>Method 2</b>		
<b>Method 3</b>		
<b>Method 4</b>		
<b>Method 5</b>		
<b>Method 6</b>		

\*Note: "Method 1" to "Method 6" means the experimental methods 1 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 6" due to insufficient table space.

Table 9: Experimental results for each pre-processing process of the 2nd experiment

Category	Method 1			Method 2			Method 3		
Epoch	60			43			67		
Accuracy	0.757432			0.765740			0.776835		
Precision	0.878238			0.865272			0.846572		
Recall	0.597735			0.629496			0.676225		
F1-score	0.711333			0.728789			0.751871		
Cohens kappa	0.514864			0.531480			0.553670		
ROC AUC	0.867342			0.867865			0.867660		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	26162	2364	Positive	25730	2796	Positive	25030	3496
	Negative	11475	17051	Negative	10569	17957	Negative	9236	19290

\*Note: "Method 1" to "Method 3" means the experimental methods 1 to 3 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 3" due to insufficient table space.

Table 10: Experimental results for each pre-processing process of the 2nd experiment

Category	Method 4			Method 5			Method 6		
Epoch	41			39			31		
Accuracy	0.758273			0.784214			0.777098		
Precision	0.820870			0.820467			0.842053		
Recall	0.660731			0.727652			0.682150		
F1-score	0.732146			0.771277			0.753714		
Cohens kappa	0.516546			0.568429			0.554196		
ROC AUC	0.848140			0.867191			0.866421		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	24413	4113	Positive	23984	4542	Positive	24876	3690
	Negative	9678	18848	Negative	7769	20757	Negative	9067	19459

\*Note: "Method 4" to "Method 6" means the experimental methods 4 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 4" to "Method 6" due to insufficient table space

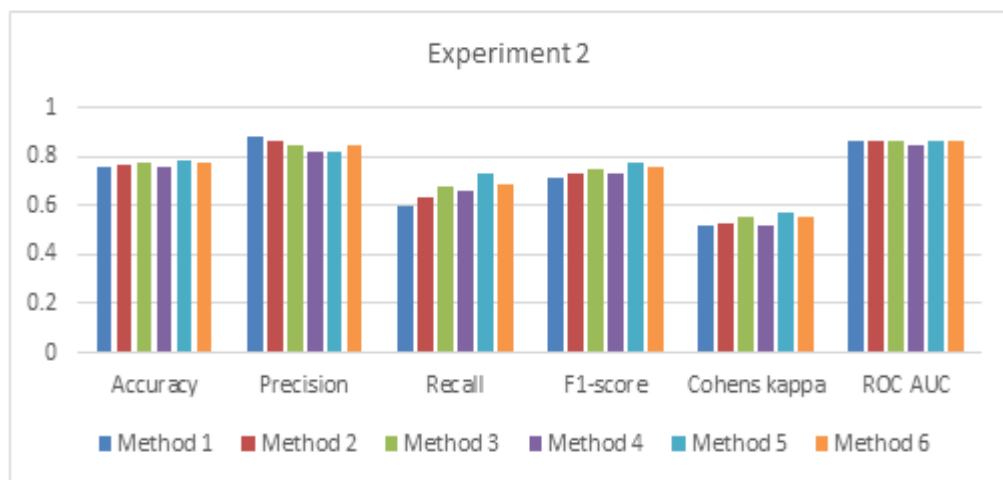
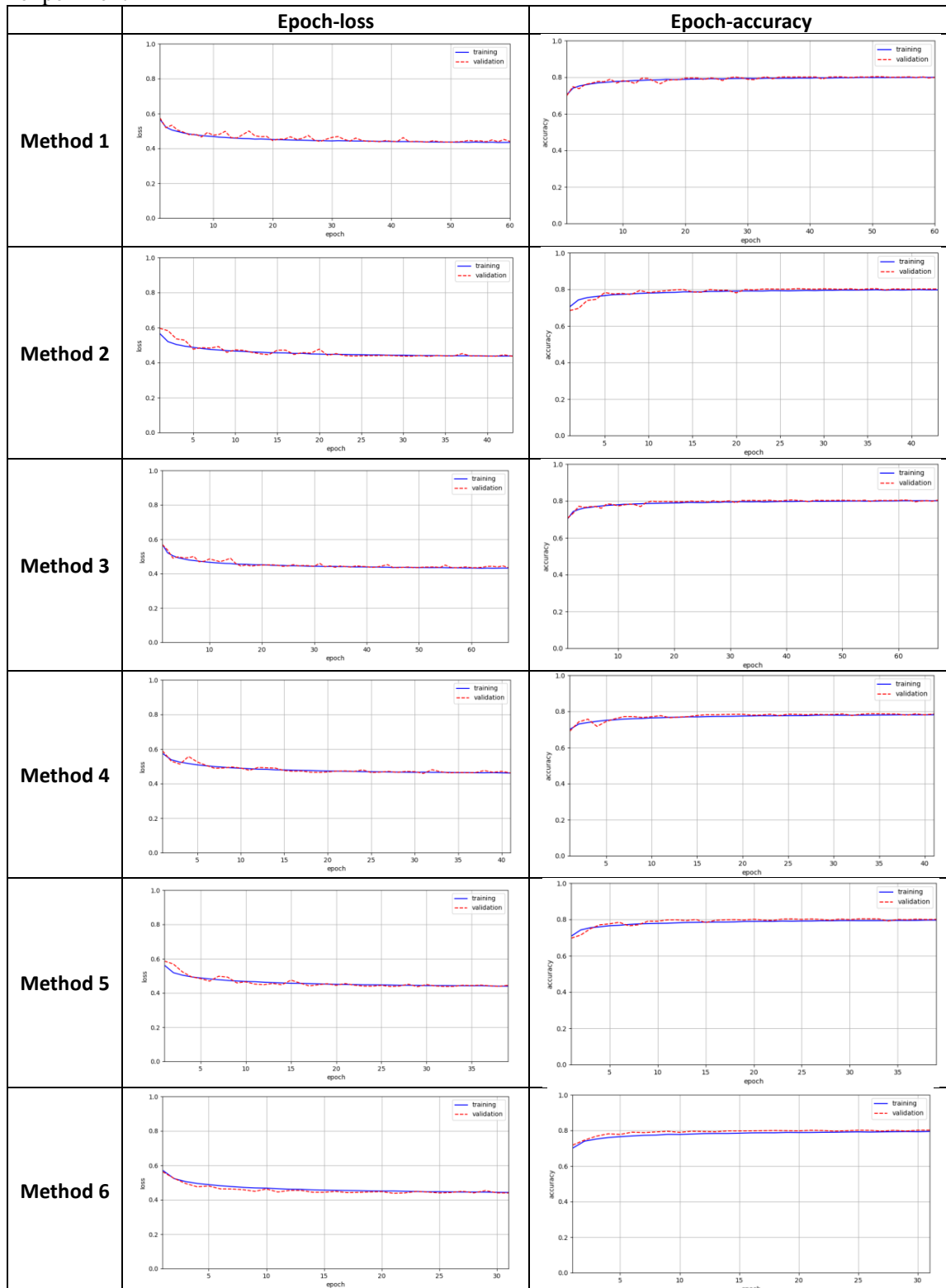


Figure 6: Comparison of all results from Experiment 2

Table 11: Visualization of the learning process for each pre-processing process of the 2nd experiment



\*Note: "Method 1" to "Method 6" means the experimental methods 1 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 6" due to insufficient table space.

Table 12: Experimental results for each pre-processing process of the 3rd experiment

Category	Method 1			Method 2			Method 3		
Epoch	56			48			38		
Accuracy	0.756503			0.762743			0.776520		
Precision	0.873469			0.871886			0.841649		
Recall	0.599909			0.615999			0.681203		
F1-score	0.711293			0.721939			0.752974		
Cohens kappa	0.513006			0.525486			0.553039		
ROC AUC	0.865633			0.865874			0.867298		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	26047	2479	Positive	25944	2582	Positive	24870	3656
	Negative	11413	17113	Negative	10954	17572	Negative	9094	19432

\*Note: "Method 1" to "Method 3" means the experimental methods 1 to 3 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 3" due to insufficient table space.

Table 13: Experimental results for each pre-processing process of the 3rd experiment

Category	Method 4			Method 5			Method 6		
Epoch	37			40			38		
Accuracy	0.756468			0.780043			0.786353		
Precision	0.825909			0.822052			0.820170		
Recall	0.649933			0.714822			0.733541		
F1-score	0.727430			0.764696			0.774441		
Cohens kappa	0.512936			0.560086			0.572706		
ROC AUC	0.846113			0.864492			0.866734		
Matrix	Actual Values	Predictive Values		Actual Values	Predictive Values		Actual Values	Predictive Values	
		Positive	Negative		Positive	Negative		Positive	Negative
	Positive	24618	3908	Positive	24112	4414	Positive	23938	4588
	Negative	9986	18540	Negative	8135	20391	Negative	7601	20925

\*Note: "Method 4" to "Method 6" means the experimental methods 4 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 4" to "Method 6" due to insufficient table space.

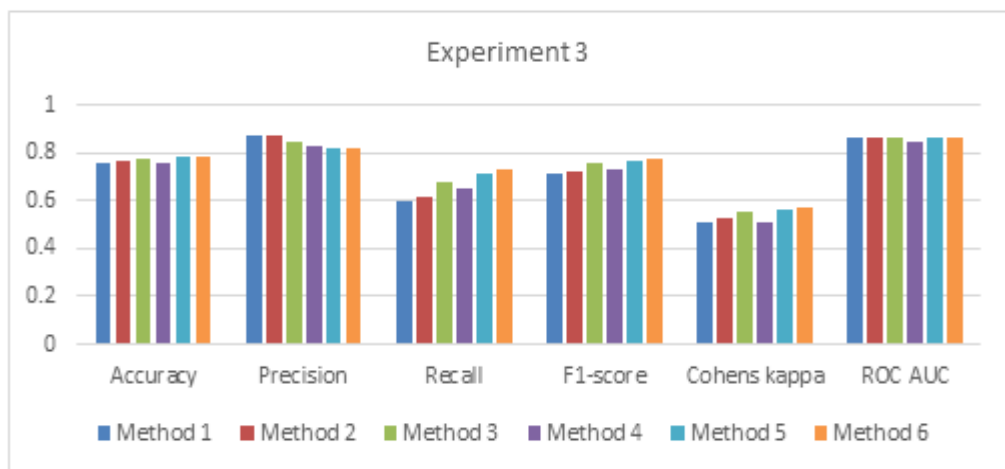
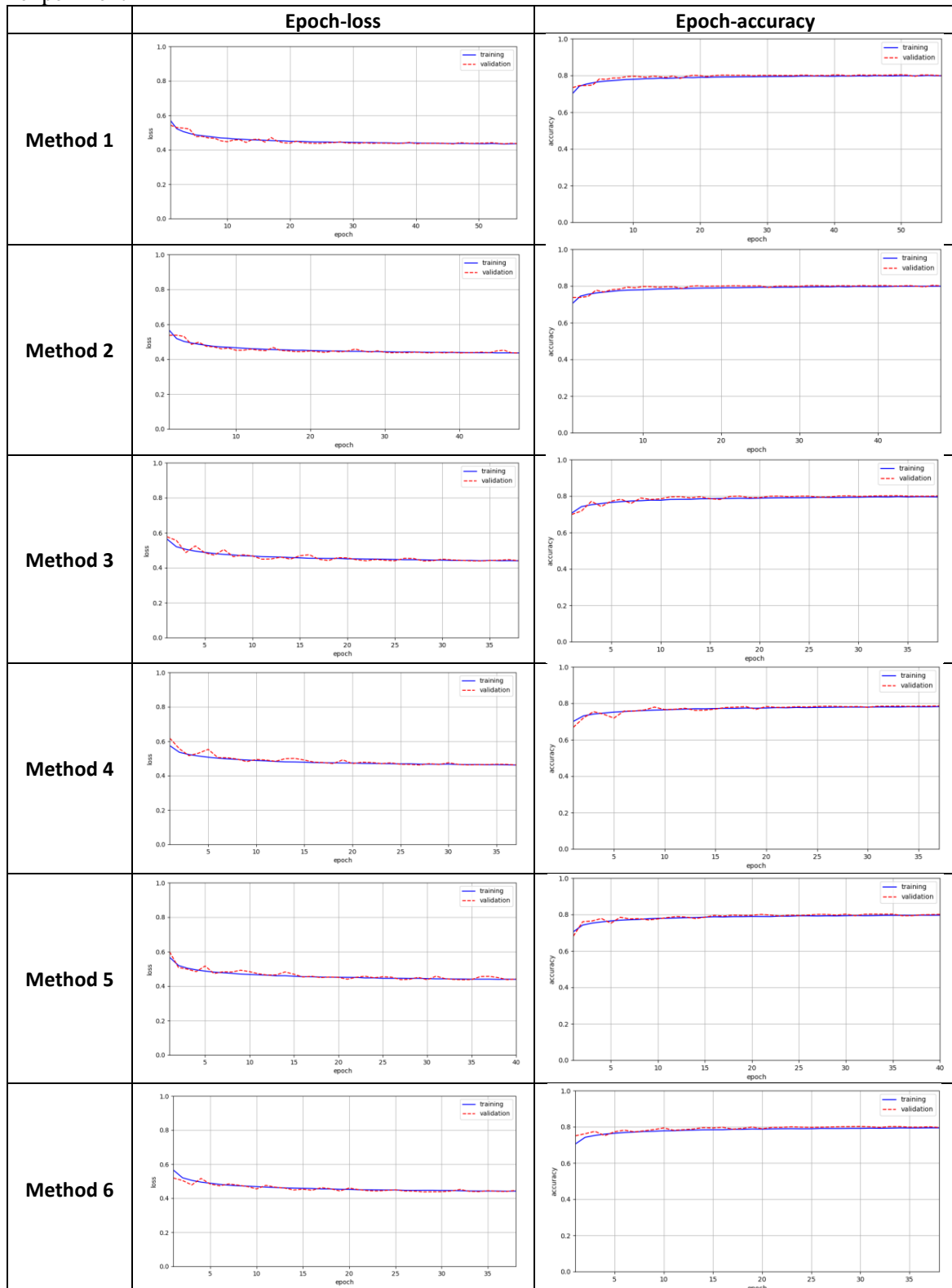


Figure 7: Comparison of all results from Experiment 3

Table 14: Visualization of the learning process for each pre-processing process of the 3rd experiment



\*Note: "Method 1" to "Method 6" means the experimental methods 1 to 6 less in "2.1 Suggest Method" in order. Marked as "Method 1" to "Method 6" due to insufficient table space.

The disadvantages of this experiment are as follows. In the case of real name, it is provided by converting it to “#@인간및인간집단.인물.유명인#” (in English, “#@human and human group.person.celebrity#”) to avoid problems. Since these stands for a different aspect from actual data, there may be differences in results when sentence data used online is used. Also, as seen from the data distribution, there is a difference in length between malicious and non-malicious texts. To make the data length as similar as possible, it was padded with a limit of 200, but since malicious articles are usually a little longer, this will have an impact. In addition, it is thought that the failure to design a CNN structure suitable for pre-processing data using the consonant and vowel separation method may influence low accuracy and high loss values. Also, to distinguish between final consonant and non-Korean combinations, a 3-token method such as { ○ } or [ ○ ], and was tested by making a dictionary and making the final and Korean combinations into different tokens during pre-processing.

In a situation where proper pre-processing such as morpheme analysis has not been performed, the accuracy of about 70% or more and less than 80% even when simple decomposition is applied is suitable for malicious texts with many variations.

This experiment can be used as a pre-processing technique for learning when new types of malicious post detection bypass techniques are developed by using the fact that data preprocessing is extremely fast. In addition, since the model is simple, it requires less computing power than complex models, so it would be good to use it for game chatting that does not require high accuracy and a large amount of text is exchanged in a brief time.

For future research, it is highly recommended to create a CNN structure suitable for preprocessing of data using each destructive method, try fine-tuning, collect more diverse Korean malicious text data such as game chat data or news comment data with similar length, look for objective and systematic criteria for judging malicious texts, and find and apply other techniques that can analyse malicious texts.

#### 4. ACKNOWLEDGEMENTS

This paper is conducted because of considering malicious text detection methods to help solve these situations after experiencing inconveniences caused by malicious text and misdetection of detection programs to prevent malicious text in online environments such as games and comments.

#### 5. References

- [1] Rules of Korean spelling. Ministry of Government Legislation Korean Law Information Center (n.d.). Retrieved Feb 14, 2023, from [https://www.law.go.kr/행정규칙/한글맞춤법/\(2017-12,20170328\)](https://www.law.go.kr/행정규칙/한글맞춤법/(2017-12,20170328))
- [2] Social and economic costs of malicious comments up to KRW 35 trillion per year. Barun ICT Research Center. (2022, December 19). Retrieved February 11, 2023, from <http://barunict.kr/?p=10678>
- [3] Report on the Survey of Cyber Violence in 2021. Korea Communications Commission. (n.d.). Retrieved February 14, 2023, from <https://kcc.go.kr/user.do?mode=view&page=A02060400&dc=K02060400&boardId=1030&cp=1&searchKey=TITLE&searchVal=사이버&boardSeq=53091>
- [4] Detecting abusive language with deep learning. NDC replay. (n.d.). Retrieved February 14, 2023, from



[http://ndcreplay.nexon.com/NDC2018/sessions/NDC2018\\_0033.html#c=NDC2018&k\[\]=](http://ndcreplay.nexon.com/NDC2018/sessions/NDC2018_0033.html#c=NDC2018&k[]=)  
 답러닝

- [5] Korean NLP in python. KoNLPy. (n.d.). Retrieved February 14, 2023, from <https://konlpy.org/en/latest/>
- [6] Morphological analysis and POS tagging. Morphological analysis and POS tagging - KoNLPy 0.6.0 documentation. (n.d.). Retrieved February 14, 2023, from <https://konlpy.org/en/latest/morph/>
- [7] Konlpy. (n.d.). 'Kkma().Nouns' raises 'jpyype.\_jexception.java.lang.outofmemoryerrorpyraisable' on a weird sentence · issue #73 · Konlpy/Konlpy. GitHub. Retrieved February 14, 2023, from <https://github.com/konlpy/konlpy/issues/73>
- [8] ⇨(internet terminology). Namuwiki. (2023, February 5). Retrieved February 14, 2023, from [https://namu.wiki/w/⇨\(인터넷 용어\)#s-1.3](https://namu.wiki/w/⇨(인터넷 용어)#s-1.3)
- [9] National Hangeul Museum Newsletter 2017. 2. Let's meet John Ross, who started the first Korean spacing. Journal of the National Hangeul Museum, February 2017, issue 43. (2017, February 21). Retrieved February 14, 2023, from <https://www.hangeul.go.kr/user/html/webzine/201702/sub2.html>
- [10] Text Ethics Verification Data. AI Hub. (2022, July 12). Retrieved February 14, 2023, from <https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=558>
- [11] Google. (n.d.). Google colaboratory. Google Colab. Retrieved February 14, 2023, from [https://colab.research.google.com/notebooks/basic\\_features\\_overview.ipynb](https://colab.research.google.com/notebooks/basic_features_overview.ipynb)
- [12] How to determine the embedding size? Artificial Intelligence Stack Exchange. (2021, July 7). Retrieved February 14, 2023, from <https://ai.stackexchange.com/questions/28564/how-to-determine-the-embedding-size>
- [13] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.
- [14] Team, K. (n.d.). Keras Documentation: CONV1D layer. Keras. Retrieved February 14, 2023, from [https://keras.io/api/layers/convolution\\_layers/convolution1d/](https://keras.io/api/layers/convolution_layers/convolution1d/)
- [15] Team, K. (n.d.). Keras Documentation: Adam. Keras. Retrieved February 14, 2023, from <https://keras.io/api/optimizers/adam/>
- [16] Twitter. (n.d.). Excluding Korea, China and Japan, the number of tweets will be expanded to 280 characters. Twitter. Retrieved February 14, 2023, from [https://blog.twitter.com/ko\\_kr/topics/product/2017/Cramming-Tweeting-Made-Easier](https://blog.twitter.com/ko_kr/topics/product/2017/Cramming-Tweeting-Made-Easier)