PHISH🐟 in MESH🕸: Korean Adversarial Phonetic Substitution and Phonetic-Semantic Feature Integration Defense

Anonymous ACL submission

Abstract

As malicious users increasingly employ phonetic substitution to evade hate speech detection, researchers have investigated such strategies. However, two key challenges remain. First, existing studies have overlooked the Korean language, despite its vulnerability to phonetic perturbations due to its phonographic nature. Second, prior work has primarily focused on constructing datasets rather than developing architectural defenses. To address these challenges, we propose (1) PHonetic-Informed Substitution for Hangul (PHISH🐟) that exploits the phonological characteristics of the Korean writing system, and (2) Mixed Encoding of Semantic-pHonetic features (MESH🕸) that enhances the detector's robustness by incorporating phonetic information at the architectural level. Our experimental results demonstrate the effectiveness of our proposed methods on both perturbed and unperturbed datasets, suggesting that they not only improve detection performance but also reflect realistic adversarial behaviors employed by malicious users.

1 Introduction

As offensive text detection systems have been advanced, malicious users have adopted more sophisticated filtering evasion strategies. In particular, they have been trying to replace characters or words in hateful texts with alternatives that are pronounced similarly. Despite requiring substantial linguistic awareness of target languages, malicious users frequently adopt phonetic substitution, which proves to be an effective method of evading detection (Boucher et al., 2022; Le et al., 2023). Therefore, researchers have formalized this strategy and proposed defense methods against it (Cooper et al., 2023; Le et al., 2022).

However, we identify two key challenges regarding the target language and the proposed defense strategies. First, existing studies on phonetic substitution attacks have rarely considered Korean. Because users of phonographic writing systems can often infer the original word from its phonetically perturbed form, such attacks may be more effective in languages like Korean (Kim, 2011). Nevertheless, prior research has largely overlooked phonetic perturbations in Korean and instead focused on language-agnostic strategies, such as inserting meaningless words (Yu et al., 2024).

Second, most of the currently proposed defense methods primarily focus on constructing perturbed datasets, while less focused on augmenting additional feature representations. Specifically, proposed defense methods often rely on fine-tuning methods using datasets specialized for each attack strategy (Lee et al., 2025). However, those approaches not only incur additional annotation costs but also raise concerns about overfitting to particular attack patterns.

To tackle these challenges, we propose (1) a PHonetic-Informed Substitution for Hangul, PHISH, and (2) sequntial or direct Mixed Encoding of Semantic-pHonetic features (seq-MESH, dir-MESH). PHISH substitutes one or two Korean unit letters per syllable with phonetically similar counterparts using the International Phonetic Alphabet (IPA) and the Korean standard pronunciation rules. Unlike prior strategies, PHISH does not use any characters or special symbols from other languages; instead, it leverages only the Korean character set. seq-MESH and dir-MESH aim to enhance the robustness of detectors against phonetic perturbation by augmenting phonetic information. Specifically, our methods adopt cross-attention mechanism to incorporate semantic and phonetic information.

To examine the effectiveness of both our proposed attack and defense methods, we conducted experiments on two Korean hate speech datasets: K-HATERS (Park et al., 2023) and KoLD (Jeong et al., 2022). Specifically, we quantified performance degradation of baseline detectors under our phonetic substitution attack. Also, we evaluated detectors equipped with our defense methods on both the original and perturbed test sets, comparing their performance to the corresponding base models. Thus, this paper has following contributions:

- We introduce a phonetic substitution attack method, PHISH, which leverages the characteristics of the Korean language and successfully misleads prior detectors.
- Also, we propose defense methods, sequential or direct MESH, which enhance the robustness of detectors by guiding them to incorporate semantic and phonetic information.

2 Attack method

Korean malicious users often circumvent filtering systems by making slight modifications to their toxic sentences. Specifically, they commonly conduct *phonetic substitution*: replacing offensive letters or words with phonetically similar alternatives. As Korean is a phonographic language with shallow orthographic depth, phonetically substituted toxic texts remain intelligible to human readers but can easily confuse detection systems that rely on semantic representations (Ellis et al., 2004). In the Korean writing system, Hangul, each character fundamentally represents a single syllable. Here, a Hangul syllable character is structured by combining individual components, called *jamo*, into a syllable block. Such a syllable block must contain at least one initial consonant (onset) and a vowel (nucleus), while a final consonant (coda) may or may not be present. For example, a Hangul syllable block '김 [kim]' consists of three jamos, onset 'ㄱ [k]', nucleus 'ㅣ [i]', and coda 'ㅁ [m].'

Based on this structural property, we propose a PHonetic-Informed Substitution for Hangul (PHISH) that perturbs Korean text to mislead detection systems. PHISH replaces a subset of jamos within each syllable with phonetically similar alternatives, using the International Phonetic Alphabet (IPA) and the Korean standard pronunciation rule. In particular, PHISH uses two degrees of attack according to the number of substituted jamos within a syllable: single-jamo attack, where only one jamo is substituted, and dual-jamo attack, where two jamos are substituted. During the attack, PHISH employs a look-up table $\mathcal{D}$ to match phonetically similar jamos. Section 2.1 details PHISH algorithm and Section 2.2 illustrates how we defined the predefined look-up table $\mathcal{D}$.

Algorithm 1 PHISH Algorithm

Input: Text $\mathcal{T} = \{\mathcal{T}_0, \cdots, \mathcal{T}_n\}$,
Perturbation ratio $r \in [0, 1]$,
Attack mode $m \in \{\text{Single}, \text{Dual}\}$
Output: Perturbed text $\mathcal{T}$

1: $\mathcal{I}_\mathcal{D}, \mathcal{I}_\mathcal{S} \leftarrow$ `Vulnerable Search`$(\mathcal{T})$
2: Shuffle $\mathcal{I}_\mathcal{D}$ and $\mathcal{I}_\mathcal{S}$
3: $n_V \leftarrow$ Total length of $\mathcal{I}_\mathcal{D}$ and $\mathcal{I}_\mathcal{S}$
4: $n_A \leftarrow 0$ ▷ # of perturbed syllables

5: **while** $\frac{n_A}{n_V} < r$ and $\mathcal{I}_\mathcal{D} \neq \varnothing$ **do**
6: Pop a target index i from $\mathcal{I}_\mathcal{D}$
7: $\mathcal{T}_i \leftarrow$ `Syllable Attack`$(\mathcal{T}_i, n_{attk})$
8: $n_A \leftarrow n_A + 1$
9: **end while**

10: **while** $\frac{n_A}{n_V} < r$ and $\mathcal{I}_\mathcal{S} \neq \varnothing$ **do**
11: Pop a target index i from $\mathcal{I}_\mathcal{S}$
12: $\mathcal{T}_i \leftarrow$ `Syllable Attack`$(\mathcal{T}_i, n_{attk})$
13: $n_A \leftarrow n_A + 1$
14: **end while**
15: **return** $\mathcal{T}$

2.1 The PHISH algorithm

Algorithm 1 shows the pseudocode of the PHISH. The algorithm takes three inputs: an input text $\mathcal{T}$, which is a sequence of syllables $\mathcal{T}_i$, a perturbation ratio r, and the degree of attack m. Here, the degrees m of 'single' and 'dual' refer to single and dual-jamo attacks, respetively.

PHISH consists of two main phases: (1) Index searching and (2) Substitution. In index searching, PHISH identifies the target indices to be perturbed (Line 1) before conducting substitution. Since some Korean syllables do not allow any perturbation because their jamos do not have any phonetically similar alternatives, the algorithm first identifies the vulnerable indices of $\mathcal{T}$ that allow our adversarial attack. Specifically, if a syllable contains more than one replaceable jamo, its index is added to $\mathcal{I}_\mathcal{D}$; otherwise, if it contains only one, the index is added to $\mathcal{I}_\mathcal{S}$. To search this index, PHISH calls 'vulnerable search' illustrated in Algorithm 2 (Section 2.1.1).

After determining the target indices, the substitution phase starts (Lines 5 to 14). In this phase, the algorithm perturbs syllables corresponding to target indices one by one until the ratio of attacked syllables reaches the given ratio r or no more vulnerable indices are left. For the substitution, PHISH uses

Algorithm 2 Vulnerable Search Algorithm

Input: Look-up table $\mathcal{D}$, Text $\mathcal{T}$
Output: Double-indices list $\mathcal{I_D}$,
Single-indices list $\mathcal{I_S}$

1: **for** each syllable $sybl$ in $\mathcal{T}$ **do**
2: $\quad c \leftarrow 0$ $\quad\triangleright$ # of substitutable jamos
3: $\quad$ **for** each jamo j in $sybl$ **do**
4: $\quad\quad$ **if** $\mathcal{D}[j] \neq \varnothing$ **then** $\triangleright$ Alternatives exist
5: $\quad\quad\quad c \leftarrow c + 1$
6: $\quad\quad$ **end if**
7: $\quad$ **end for**

8: $\quad$ **if** $c \geq 2$ **then**
9: $\quad\quad$ Add the index of $sybl$ into $\mathcal{I_D}$
10: $\quad$ **else if** $c = 1$ **then**
11: $\quad\quad$ Add the index of $sybl$ into $\mathcal{I_S}$
12: $\quad$ **end if**
13: **end for**
14: **return** $\mathcal{I_D}$ and $\mathcal{I_S}$

Algorithm 3 Syllable Attack Algorithm

Input: Look-up table $\mathcal{D}$, Syllable $sybl$,
Degree of attack $m \in$ {Single, Dual}
Output: Perturbed syllable $sybl$

1: Initialize # of substitutable jamos n_{sttd} as 0
2: **if** $m =$ Single **then** $n_{attk} \leftarrow 1$
3: **else if** $m =$ Dual **then** $n_{attk} \leftarrow 2$
4: **end if**
5: Decompose $sybl$ into a list of jamos $\mathcal{J}$.
6: Shuffle list $\mathcal{J}$.

7: **for** each jamo j in $\mathcal{J}$ **do**
8: $\quad$ **if** $\mathcal{D}[j] \neq \varnothing$ **then**
9: $\quad\quad$ Substitute j with random jamo in $\mathcal{D}[j]$
10: $\quad\quad n_{sttd} \leftarrow n_{sttd} + 1$
11: $\quad$ **end if**
12: $\quad$ **if** $n_{sttd} = n_{attk}$ **then**
13: $\quad\quad$ **break**
14: $\quad$ **end if**
15: **end for**
16: Recompose $sybl$ with substituted jamos $\mathcal{J}$
17: **return** $sybl$

syllable attack algorithm, which is illustrated in Section 2.1.2. After the substitution phase is done, PHISH returns the perturbed text $\mathcal{T}$.

2.1.1 Vulnerable Search Algorithm

Algorithm 2 shows the search algorithm for identifying target indices of a given text $\mathcal{T}$. When $\mathcal{T}$ is inputted, the algorithm checks whether each syllable $sybl$ in $\mathcal{T}$ allows perturbation. In detail, the algorithm iterates over each syllable in $\mathcal{T}$, and checks whether each jamo composing each syllable has alternatives by referring to a predefined look-up table (Lines 2 to 7). When a syllable has substitutable jamo, the index of syllable is appended to $\mathcal{I_D}$ or $\mathcal{I_S}$ according to the number of substitutable jamos (Lines 8 to 12). After the iteration, the algorithm returns the two indices list, $\mathcal{I_D}$ and $\mathcal{I_S}$.

2.1.2 Syllable Attack Algorithm

Algorithm 3 illustrates the process of attack syllables. The algorithm requires a syllable to be attacked and the degree of attack. After deciding the number of jamos to be substituted (Lines 2 to 3), we decompose the inputted syllable $sybl$ into a list of jamos (Line 5). Then, the decomposed list is shuffled to substitute jamos with a random order. After, the algorithm substitutes each jamo with its phonetically similar alternatives by using the look-up table $\mathcal{D}$ until the number of substituted jamos n_{stt} reaches the predefined threshold n_{attk} (Lines 7 to 14). After the substitution, the algorithm returns the perturbed syllable, composed of substituted jamos.

2.2 Look-up Table for Alternatives

Our proposed adversarial attack, PHISH, requires a predefined look-up table $\mathcal{D}$ that maps a jamo to a set of phonetically similar jamos. As previously mentioned, a Korean syllable consists of an initial consonant, a medial vowel, and an optional final consonant. Thus, we applied different procedures for each component of syllables when constructing the look-up table. Appendix A illustrates the table.

To classify initial consonants, we used their base IPA symbols as the guiding principle. In Korean, some initial consonants share the same place and manner of articulation. We grouped initial consonants sharing similar articulatory features or the base phone. For example, 'ㅂ [p]' and 'ㅍ [p^h]' are variants of the base phone [p]'. While their differences arise from laryngeal settings, such distinctions contribute less to phonetic similarity than their articulation place and manner. Accordingly, we defined five sets for the initial consonants regarding their base phone.

When defining the table for final consonants, we used the Korean standard pronunciation rule as the principle. Unlike initial consonants, which

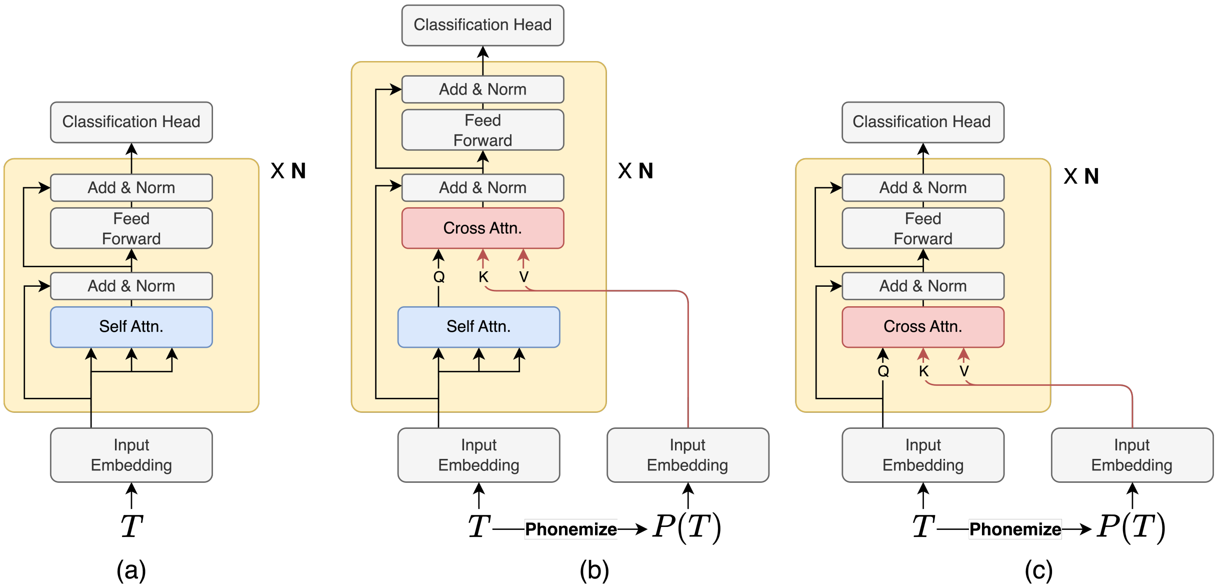

Figure 1: Architectures of base models and our methods. (a) shows the architecture of base detectors using the self-attention mechanism; (b) shows the architecture of seq-MESH detectors using stacked self and cross-attention layers; (c) shows the architecture of dir-MESH detectors using cross-attention instead of self-attention.

are pronounced distinctly from others, some final consonants are pronounced identically according to the Korean standard pronunciation rule. As this phonological property is aligned with the motivation of PHISH, we directly used this rule to define the table for final consonants. Following this procedure, we defined six jamo sets corresponding to one of the six standard pronunciations.

To classify the medial vowels, we grouped monophthongs and diphthongs that share the same base phone. Specifically, there are 11 Korean diphthongs that are derived from monophthongs by combining them with glides such as '/w/' or '/j/.' Since diphthongs and their corresponding base vowels are pronounced in a similar way, we grouped them into the same substitution sets. Accordingly, we defined seven sets of vowels.

3 The MESH Defense Methods

We hypothesized that augmenting the phonetic information can enhance the robustness of detectors regarding two aspects against phonetic substitution attacks: (1) providing supplementary features, and (2) mitigating information loss. First, the augmented phonetic information allows prior detectors, which rely on semantic-level representations, to leverage alternative linguistic cues during their detection. Since semantics of perturbed texts can significantly deviate from that of original text, such phonetic cues can help to address such deviation.

Second, augmenting phonetic features can mitigate the information loss caused by perturbations. As perturbation increases the likelihood of unknown tokens during the tokenization process of detectors, it can severely disrupt the semantic structure of the text (Yu et al., 2024). This disruption can interfere with the detectors' semantic understanding of the text. As phonetic information can provide a hint for reconstructing the unknown word, we believe that augmenting phonetic information can recover such information loss.

To combine phonetic information to detectors, we propose sequential or direct Mixed Encoding of Semantic-pHonetic features (in short, seq-MESH or dir-MESH) that use additional or alternative cross-attention layers. Figure 1 compares prior detectors and our proposed methods. Previously, detectors use self-attention layers to process or capture the semantic meaning of the input text, as shown in Figure 1 (a). Meanwhile, seq-MESH and dir-MESH combine the input text and its phoneme sequence using cross-attention layers, as shown in Figure 1 (b) and (c). Here, to generate phoneme sequences, we used a widely adopted open-source Korean phonemizer[1].

3.1 Sequential MESH

While seq-MESH follows the same overall architecture of previous detectors, it differs by incorpo-

[1] https://github.com/Kyubyong/g2pK

rating an additional cross-attention layer in every encoder block. This additional layer computes attention between the semantics of the input text and its phoneme sequence. Specifically, we used the output of the preceding self-attention output as query; and the embedded phoneme sequence is used for key and value. As a result, seq-MESH can fully leverage and incorporate the two different types of features using two attention layers.

3.2 Direct MESH

Since self-attention layers specialize in capturing and processing the semantics of the text, they may propagate the distorted semantics caused by unknown tokens in perturbed texts. Specifically, this error propagation of self-attention layers can affect the subsequent layers and mislead the detectors. To address this, we further propose dir-MESH, which replaces the self-attention layers with cross-attention layers to incorporate semantic and phonetic information directly, while relaxing the possibility of error propagation. Specifically, we use the same architecture with seq-MESH except for self-attention layers.

4 Experiment

4.1 Datasets

We used two Korean hate speech datasets for our experiment: K-HATERS (Park et al., 2023) and KoLD (Jeong et al., 2022). Both datasets used online comments to crawl hate speech and labeled them. Specifically, K-HATERS used a Korean online news platform as the source. They labeled hate speech into various sub-categories, regarding the intensity of hatefulness. Similarly, KoLD crawled the same platform and YouTube to construct the dataset. KoLD used labels different from K-HATERS for offensive samples.

As our study aims to examine the effectiveness of phonetic methods on hateful speech, we decided to focus on coarse labels: *offensive* or *normal*. Though two datasets provided detailed labels, we gathered fine-grained offensive labels into a single category. Since this gathering process produced highly imbalanced regarding these two labels, we downsampled the datasets. Consequently, we used 104,112 samples from K-HATERS and 40,429 samples from KoLD. These samples are split into training, validation, and test sets with a ratio of 8:1:1.

After collecting datasets for our experiment, we collected additional perturbed test sets. Using PHISH, we derived different test sets with different settings, including attack ratios and degrees. Specifically, we conducted attacks under three perturbation ratios (10, 20, and 30%) and two degrees of attack (single-jamo and dual-jamo). Note that we did not alter training set; all methods are trained on the original data without applying PHISH.

4.2 Baselines and MESH variants

For baselines, we used three small language models that are commonly used in prior Korean hate speech detection research and adopt the self-attention mechanism: KLUE-BERT, KLUE-RoBERTa (Park et al., 2021b), and KCBERT (Lee, 2020). These three models possess Korean language understanding capabilities. Specifically, KLUE-BERT and RoBERTa was pretrained on KLUE dataset (Park et al., 2021b), which is a Korean language understanding benchmark. Meanwhile, KCBERT was primarily trained on web-based data such as news articles and user comments. So, it tends to exhibit stronger baseline performance in tasks related to hate-speech detection compared to the other two.

For implementing detectors equipped with seq-MESH or dir-MESH, we reused the parameters as in Rothe et al. (2020). We set initial parameters of two methods by copying that of the base models, rather than initializing from scratch. Specifically, the self-attention weights of the base models were copied into cross-attention weights of two methods.

4.3 Environment of Experiment

We used a single RTX A6000 for training and evaluating the models. We trained each model for five epochs with a learning rate of 10^{-5} and a batch size of 32. Then, we chose checkpoints with the highest F1 score on the validation set. We repeated each dataset experiment 10 times with different random seeds to ensure reproducibility.

5 Result and Discussion

In this section, we present our experimental results, which are shown in Tables 1, 2, 3, and 4. Tables display the average and standard deviation of F1 scores across the ten experiments. We found three findings of our methods: (1) degradation of performance under PHISH, (2) robustness of seq-MESH and dir-MESH against the attack scenario, and (3) the alignment between real-world scenarios and our attack and defense methods.

First, we quantified the performance degradation of three base models under PHISH using dif-

Attack Ratio	0%	10%		20%		30%	
	F1	F1	ΔF1	F1	ΔF1	F1	ΔF1
BERT	73.8±0.2	73.6±0.3	-0.2±0.4	71.9±0.3	-1.9±0.4	69.5±0.3	-4.3±0.4
RoBERTa	65.0±2.0	63.5±2.6	-1.5±3.3	58.2±3.3	-6.8±3.9	52.4±4.6	-12.6±5.0
KCBERT	76.2±0.4	75.7±0.3	-0.5±0.5	74.8±0.3	-1.4±0.5	73.4±0.3	-2.8±0.5
BERT$_{\text{dir-MESH}}$	74.2±0.5	73.1±0.4	-1.1±0.6	71.7±0.5	-2.5±0.7	70.2±0.8	-4.0±0.9
RoBERTa$_{\text{dir-MESH}}$	74.4±0.4	72.9±0.5	-1.3±0.6	71.2±0.8	-3.2±0.9	69.9±0.7	-4.5±0.8
KCBERT$_{\text{dir-MESH}}$	76.6±0.4	75.5±0.5	-1.1±0.6	74.2±0.5	-2.4±0.6	73.2±0.6	-3.4±0.7
BERT$_{\text{seq-MESH}}$	78.9±0.4	76.2±0.5	-2.7±0.6	73.0±0.5	-5.9±0.6	70.8±0.4	-8.1±0.6
RoBERTa$_{\text{seq-MESH}}$	74.6±0.6	73.4±0.6	-1.2±0.8	71.4±0.8	-3.2±1.0	70.1±0.9	-4.5±1.1
KCBERT$_{\text{seq-MESH}}$	**80.8±0.2**	**79.2±0.3**	-1.6±0.4	**77.8±0.3**	-3.0±0.3	**74.9±0.4**	-5.9±0.4

Table 1: Detection performance on K-HATERS dataset with single-jamo attack

Attack Ratio	0%	10%		20%		30%	
	F1	F1	ΔF1	F1	ΔF1	F1	ΔF1
BERT	75.1±0.5	74.6±0.6	-0.5±0.8	70.6±0.7	-4.5±0.9	66.4±1.5	-8.7±1.6
RoBERTa	72.6±1.6	71.6±1.7	-1.0±2.3	66.6±3.0	-6.0±3.4	60.8±5.8	-11.8±6.0
KCBERT	77.5±0.4	76.7±0.6	-0.8±0.7	75.4±0.7	-2.1±0.8	72.6±1.3	-4.9±1.4
BERT$_{\text{dir-MESH}}$	75.9±0.5	75.0±0.7	-0.9±0.9	73.0±0.6	-2.9±0.8	71.6±0.8	-4.3±0.9
RoBERTa$_{\text{dir-MESH}}$	75.9±0.7	75.1±0.6	-0.8±0.9	73.6±0.4	-2.3±0.8	71.3±0.6	-4.6±0.9
KCBERT$_{\text{dir-MESH}}$	77.7±0.5	76.8±0.6	-0.9±0.8	74.9±0.6	-2.8±0.8	73.9±0.6	-3.8±0.8
BERT$_{\text{seq-MESH}}$	79.5±1.0	77.9±0.9	-1.6±1.3	74.8±0.9	-4.7±1.3	73.0±0.9	-0.65±1.3
RoBERTa$_{\text{seq-MESH}}$	75.9±0.5	74.9±0.3	-1.0±0.6	73.3±0.4	-2.6±0.6	71.4±0.5	-4.5±0.7
KCBERT$_{\text{seq-MESH}}$	**81.4±0.5**	**80.3±0.6**	-1.1±0.8	**78.9±0.7**	-2.5±0.9	**76.2±0.7**	-5.2±0.9

Table 2: Detection performance on KoLD dataset with single-jamo attack

ferent attack settings to validate its effectiveness. The experimental result shows that the F1 scores of all base models declined approximately as the attack ratio increased, regardless of the dataset. Specifically, with the 30% attack ratio using single-jamo attack, KCBERT's F1 scores decreased by 2.8 and 4.9 points on the K-HATERS and KoLD datasets, respectively, while BERT and RoBERTa showed larger drops ranging from 4.3 to 12.6 points. Moreover, since the dual-jamo attack perturbs more jamos per syllable than the single-jamo attack, it led to greater performance degradation on the perturbed datasets. For instance, with a 20% attack ratio, BERT and RoBERTa showed F1 score drops on the K-HATERS dataset of 4.2 and 20.1, respectively. These degradations are significantly larger than the 1.9 and 6.8 decrement observed under the single-jamo attack with the same attack ratio.

We suspect this effectiveness stems from the semantic distortion that PHISH made. Specifically, PHISH may increase the likelihood of unknown tokens during the tokenization process in detectors, which can lead to the omission of the semantic content of texts. Also, in some cases, the perturbed syllables may have been converted into homophones, which could have partially altered the semantic interpretation of the sentence. Appendix B details the statistics of unknown tokens of tokenized texts of each detector and provides additional discussion.

Second, we compared the performance of detectors using seq-MESH or dir-MESH with their corresponding base models on perturbed test sets. While base models struggled to identify perturbed offensive texts, detectors incorporating seq-MESH or dir-MESH consistently outperformed their base counterparts. This trend became more pronounced as the perturbation ratio or attack degree increased. For example, when the KoLD dataset was attacked with a 10% single-jamo perturbation, the performance gaps between the base BERT (74.6%) and its dir-MESH and seq-MESH variants (75.0 and 77.9) were 0.4 and 3.3 F1 points, respectively. Un-

Attack Ratio	0%	10%		20%		30%	
	F1	F1	ΔF1	F1	ΔF1	F1	ΔF1
BERT	73.8±0.2	73.1±0.4	-0.7±0.4	69.6±0.4	-4.2±0.4	66.3±0.7	-7.5±0.7
RoBERTa	65.0±2.0	58.2±4.0	-6.8±4.5	44.9±6.2	-20.1±6.5	31.5±7.6	-33.5±7.9
KCBERT	76.2±0.4	75.1±0.3	-1.1±0.5	72.5±0.2	-3.7±0.4	70.6±0.3	-5.6±0.5
BERT$_{\text{dir-MESH}}$	74.2±0.5	72.6±0.6	-1.6±0.8	69.7±0.6	-4.5±0.8	67.6±1.0	-6.6±1.1
RoBERTa$_{\text{dir-MESH}}$	74.4±0.4	72.2±0.7	-2.2±0.8	68.9±0.8	-5.5±0.9	67.3±1.0	-7.1±1.1
KCBERT$_{\text{dir-MESH}}$	76.6±0.4	74.9±0.3	-1.7±0.5	72.4±0.5	-4.2±0.6	71.1±0.6	-5.5±0.7
BERT$_{\text{seq-MESH}}$	78.9±0.4	75.5±0.4	-3.4±0.6	71.9±0.4	-7.0±0.6	69.6±0.8	-9.3±0.9
RoBERTa$_{\text{seq-MESH}}$	74.6±0.6	72.8±0.6	-1.8±0.8	69.7±0.8	-4.9±1.0	67.7±0.9	-6.9±1.1
KCBERT$_{\text{seq-MESH}}$	**80.8±0.2**	**77.7±0.3**	-3.1±0.4	**73.8±0.4**	-7.0±0.4	**71.6±0.7**	-9.2±0.7

Table 3: Detection performance on K-HATERS dataset with dual-jamo attack

Attack Ratio	0%	10%		20%		30%	
	F1	F1	ΔF1	F1	ΔF1	F1	ΔF1
BERT	75.1±0.5	73.5±0.7	-1.6±0.9	67.2±1.7	-7.9±1.8	56.6±3.5	-18.5±3.5
RoBERTa	72.6±1.6	69.7±2.5	-2.9±3.0	56.5±8.3	-16.1±8.5	41.6±13.6	-31.0±13.7
KCBERT	77.5±0.4	76.2±0.5	-1.3±0.6	74.0±1.0	-3.5±1.1	69.2±2.5	-8.3±2.5
BERT$_{\text{dir-MESH}}$	75.9±0.5	74.5±0.9	-1.4±1.0	71.1±1.0	-4.8±1.1	68.9±1.3	-7.0±1.4
RoBERTa$_{\text{dir-MESH}}$	75.9±0.7	74.3±0.7	-1.6±1.0	70.8±0.7	-5.1±1.0	69.6±0.9	-6.3±1.1
KCBERT$_{\text{dir-MESH}}$	77.7±0.5	76.0±0.7	-1.7±0.9	74.1±0.5	-3.6±0.7	72.4±0.8	-5.3±0.9
BERT$_{\text{seq-MESH}}$	79.5±1.0	77.6±1.3	-1.9±1.6	73.7±1.4	-5.8±1.7	70.9±2.3	-8.6±2.5
RoBERTa$_{\text{seq-MESH}}$	75.9±0.5	74.6±0.4	-1.3±0.6	70.6±0.8	-5.3±0.9	69.3±0.8	-6.6±0.9
KCBERT$_{\text{seq-MESH}}$	**81.4±0.5**	**79.6±0.5**	-1.8±0.7	**76.4±0.7**	-0.5±0.9	**72.7± 0.7**	-8.7±0.9

Table 4: Detection performance on KoLD dataset with dual-jamo attack

der a stronger 30% dual-jamo attack, these gaps increased to 13.3 and 14.3 points: 56.6, 68.9, and 70.9% for those three models.

These results indicate that our defense methods enhance robustness against phonetic perturbations since they use complementary information. Such complementary information is not only useful in recovering semantic loss but also improving the overall detection performance. Specifically, KCBERT$_{\text{seq-MESH}}$ outperformed other models including its base model, though KCBERT had already been pretrained on online comments and exhibited strong baseline performance. We believe that such further improvement demonstrates complementary benefits of our methods.

Lastly, we tested whether our methods realistically capture perturbations observed in real-world data. By evaluating their performance on original test sets (0% attack), the result showed that seq-MESH showed higher performance than their corresponding base models. Specifically, on KoLD dataset, KCBERT$_{\text{seq-MESH}}$ achieved 81.4% F1 score, which is 3.9% higher than its base model.

These improvements indicate that our assumption of phonetic perturbation is present in the real world. We assumed that malicious users adopt phonetic substitutions to deceive detectors. And, the improvement of our defense methods on original test sets supports this; the real-world dataset may contain such phonetic substitutions, as our method improves the detection performance. So, we conclude that our methods seem to align with the strategies of real-world malicious users.

6 Background

6.1 Textual Perturbation Attack

As malicious users have been attempting to conduct more sophisticated filtering evasion methods, such as visual or phonetic substitutions, researchers have attempted to formalize such strategies (Aggarwal and Zesch, 2022; Puertas and Martinez-Santos, 2021). For example, Aggarwal and Zesch (2022) summarized 12 obfuscation strategies based on a user study and analyzed the impact of these strategies across diverse datasets using ten detection models. Puertas and Martinez-Santos (2021) profiled hate speech spreaders using the frequencies of lexical and phonetic features from their texts.

Since such adversarial attacks are not universally applicable across all languages due to differences in features such as writing systems, it is crucial to account for language-specific constraints. For example, visual substitution strategies are not applicable to the Korean language because Unicode encoding does not support the replacement of Hangul jamo with visually-similar non-Hangul characters. So, researchers have investigated more language-specific adversarial attacks designed explicitly for the Korean language system (Park et al., 2021a; Perea and Lupker, 2004; Yu et al., 2024). For example, to reflect the diverse forms of offensive language used by real-world users, Park et al. (2021a) augments training data by using multiple tokenizers. Yu et al. (2024) proposed adversarial attack strategies, such as inserting, copying, and decomposing, that are commonly adopted by Korean malicious users. However, these studies did not explore phonetic substitution despite its effectiveness and applicability, as we verified in our experiment.

6.2 Defense Against Textual Perturbations

To defend against textual perturbations conducted by malicious users, researchers have proposed strategy-specific datasets (Cooper et al., 2023; Lee et al., 2025; Seth et al., 2023; Laboreiro and Oliveira, 2014) or model architectural methods. Regarding datasets, Laboreiro and Oliveira (2014) curated a profanity-annotated dataset from Portuguese online comments, identifying 17 obfuscation strategies including phonetic and symbolic substitutions. Also, Lee et al. (2025) constructed a phishing email dataset incorporating visual perturbations and demonstrated a detection method using CharacterBERT (El Boukkouri et al., 2020). However, these methods require manually crafted datasets to train defense methods. Also, fine-tuning on a specific perturbation may cause overfitting on the perturbation. Meanwhile, our defense method took different approach from these studies. Specifically, our method do not require any additional datasets for phonetic perturbations; rather, we showed that training on a real-world training set without any phonetic attack is enough to achieve good detection performance.

Some researchers have aimed to propose defense methods in perspective of detector architecture (Yang and Lin, 2020; Yu et al., 2024; Shekhar and Venkatesan, 2018; Yi et al., 2021). For example, Yu et al. (2024) leveraged layer pooling methods to enhance the robustness of detectors against textual perturbations. Yi et al. (2021) proposed an embedding model to address misbehaviors of detectors caused by morphologically similar words. Since these approaches rely solely on input text, they may lack robustness against phonetic perturbations that cause semantic distortion. In contrast, our defense address semantic distortion by supplementing the input with phonetic features. Enabling detectors to integrate them as additional information, our method demonstrated strong performance gain.

7 Conclusion

In this paper, we suggested PHISH, a phonetic substitution attack method tailored for the Korean language. Also, we proposed MESH, two defense mechanisms designed to enhance robustness against such phonetic perturbations. PHISH exploits the structural and phonographic characteristics of Hangul; the attack method substitutes one or two jamos per syllable with phonetically similar alternatives, using a predefined IPA-based look-up table. Meanwhile, our defense methods incorporate phoneme-level features through cross-attention mechanisms to integrate semantic representations with phonetic information.

Experimental results on two Korean hate speech datasets demonstrated the effectiveness of PHISH in degrading the performance of baseline detectors, validating its adversarial potential. Furthermore, detectors equipped with seq-MESH or dir-MESH consistently outperformed their base models across both perturbed and original test sets, suggesting that our defense methods not only improve robustness but also can be generalized to real-world data where phonetic substitutions may naturally occur.

These findings suggest that phonetic perturbation is a practically relevant and realistic threat in Korean text processing, and that integrating phonetic information into model architectures can mitigate semantic distortion and thus improve detection performance. We hope our work encourages further exploration of language-specific perturbation strategies and architectural defenses that go beyond dataset-level solutions.

8 Limitations

Despite the effectiveness of our methods, this paper has three limitations. First, PHISH may not be universally applicable across all languages. Specifically, PHISH is designed under the assumption that human readers can easily infer the original text from its perturbed form. As previously discussed, this assumption generally holds in languages with shallow orthographic depth, such as Korean, but may not hold in languages with deeper orthographic systems.

Second, seq-MESH and dir-MESH are inherently tied to transformer-based architectures that rely on attention mechanisms. This architectural dependence limits the applicability of our defense methods to models without self-attention, such as CNNs (Krizhevsky et al., 2012) or traditional RNN-based classifiers. In addition, integrating phoneme-level information through additional cross-attention mechanism introduces computational overhead, which may hinder deployment in resource-constrained environments.

Lastly, the effectiveness of seq-MESH and dir-MESH requires an external phonemizer to generate phoneme sequences. This means that the accuracy of such a phonemizer can affect the performance of our defense methods. However, since we used the phonemizer without any optimization or refinement, we believe the reported performance represented in our paper could be improved by using a more accurate phonemizer.

References

Piush Aggarwal and Torsten Zesch. 2022. Analyzing the real vulnerability of hate speech detection systems against targeted intentional noise. In *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*, pages 230–242, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004.

Portia Cooper, Mihai Surdeanu, and Eduardo Blanco. 2023. Hiding in plain sight: Tweets with hate speech masked by homoglyphs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2922–2929, Singapore. Association for Computational Linguistics.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nick C Ellis, Miwa Natsume, Katerina Stavropoulou, Lorenc Hoxhallari, Victor HP Van Daal, Nicoletta Polyzoe, MARIA-LOUISA TSIPA, and Michalis Petalas. 2004. The effects of orthographic depth on learning to read alphabetic, syllabic, and logographic scripts. *Reading research quarterly*, 39(4):438–468.

Younghoon Jeong, Juhyun Oh, Jongwon Lee, Jaimeen Ahn, Jihyung Moon, Sungjoon Park, and Alice Oh. 2022. KOLD: Korean offensive language dataset. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10818–10833, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Young-Suk Kim. 2011. Considering linguistic and orthographic features in early literacy acquisition: Evidence from korean. *Contemporary Educational Psychology*, 36(3):177–189.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Gustavo Laboreiro and Eugénio Oliveira. 2014. What we can learn from looking at profanity. In *Computational Processing of the Portuguese Language: 11th International Conference, PROPOR 2014, São Carlos/SP, Brazil, October 6-8, 2014. Proceedings 11*, pages 108–113. Springer.

Thai Le, Jooyoung Lee, Kevin Yen, Yifan Hu, and Dongwon Lee. 2022. Perturbations in the wild: Leveraging human-written text perturbations for realistic adversarial attack and defense. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2953–2965, Dublin, Ireland. Association for Computational Linguistics.

Thai Le, Yiran Ye, Yifan Hu, and Dongwon Lee. 2023. Cryptext: Database and interactive toolkit of human-written text perturbations in the wild. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 3639–3642.

Hanyong Lee, Chaelyn Lee, Yongjae Lee, and Jaesung Lee. 2025. BitAbuse: A dataset of visually perturbed texts for defending phishing attacks. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4367–4384, Albuquerque, New Mexico. Association for Computational Linguistics.

Junbum Lee. 2020. Kcbert: Korean comments bert. In *Proceedings of the 32nd Annual Conference on Human and Cognitive Language Technology*, pages 437–440.

Chaewon Park, Suhwan Kim, Kyubyong Park, and Kunwoo Park. 2023. K-haters: A hate speech detection corpus in korean with target-specific ratings. *Findings of the EMNLP 2023*.

San-Hee Park, Kang-Min Kim, Seonhee Cho, Jun-Hyung Park, Hyuntae Park, Hyuna Kim, Seongwon Chung, and SangKeun Lee. 2021a. KOAS: Korean text offensiveness analysis system. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 72–78, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Ji Yoon Han, Jangwon Park, Chisung Song, Junseong Kim, Youngsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Lucy Park, Alice Oh, Jung-Woo Ha (NAVER AI Lab), Kyunghyun Cho, and Kyunghyun Cho. 2021b. Klue: Korean language understanding evaluation. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Manuel Perea and Stephen J Lupker. 2004. Can caniso activate casino? transposed-letter similarity effects with nonadjacent letter positions. *Journal of memory and language*, 51(2):231–246.

Edwin Puertas and Juan Carlos Martinez-Santos. 2021. Phonetic detection for hate speech spreaders on twitter.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Dev Seth, Rickard Stureborg, Danish Pruthi, and Bhuwan Dhingra. 2023. Learning the legibility of visual text perturbations. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3260–3273, Dubrovnik, Croatia. Association for Computational Linguistics.

Ankita Shekhar and M. Venkatesan. 2018. A bag-of-phonetic-codes modelfor cyber-bullying detection in twitter. In *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pages 1–7.

Hsu Yang and Chuan-Jie Lin. 2020. TOCP: A dataset for Chinese profanity processing. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 6–12, Marseille, France. European Language Resources Association (ELRA).

MoungHo Yi, MyungJin Lim, Hoon Ko, and JuHyun Shin. 2021. Method of profanity detection using word embedding and lstm. *Mobile Information Systems*, 2021(1):6654029.

Seunguk Yu, Juhwan Choi, and YoungBin Kim. 2024. Don‘t be a fool: Pooling strategies in offensive language detection from user-intended adversarial attacks. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3456–3467, Mexico City, Mexico. Association for Computational Linguistics.

Type	Base	Jamo set
Onset	/k/	{ ㄱ, ㄲ, ㅋ }
	/t/	{ ㄷ, ㄸ, ㅌ }
	/p/	{ ㅂ, ㅃ, ㅍ }
	/tɕ/	{ ㅈ, ㅉ, ㅊ }
	/s/	{ ㅅ, ㅆ }
Nucleus	/i/	{ ㅣ, ㅢ }
	/u/	{ ㅜ, ㅠ }
	/o/	{ ㅗ, ㅛ }
	/ʌ/	{ ㅓ, ㅝ, ㅕ }
	/a/	{ ㅏ, ㅘ, ㅑ }
	/e/	{ ㅔ, ㅞ, ㅖ }
	/ɛ/	{ ㅐ, ㅙ, ㅒ }
Coda	/k/	{ ㄱ, ㄲ, ㅋ, ㄳ, ㄺ }
	/n/	{ ㄴ, ㄵ, ㄶ }
	/t/	{ ㅅ, ㅆ, ㄷ, ㅌ, ㅈ, ㅊ, ㅎ }
	/l/	{ ㄹ, ㄺ, ㄼ, ㄽ, ㄾ, ㅀ }
	/m/	{ ㅁ, ㄻ }
	/p/	{ ㅂ, ㅍ, ㄼ, ㅄ, ㄿ }

Table 5: Predefined look-up table

A Look-up table

Table 5 illustrates the predefined look-up table for Korean initial consonants (onset), vowels (nucleus), and final consonants (coda). Jamos assigned to the same set can be substituted with others in the same set. Each IPA symbol of the initial consonants (onset) and vowels (nucleus) indicates the base phone of the corresponding jamo set. Additionally, final consonants (coda) are pronounced as their corresponding base phones according to the Korean standard pronunciation rule.

B Statistics of Texts

Tables 6 and 7 on page 11 present the appearance rates of unknown tokens in both text and phoneme sequences across different detectors after conducting our PHISH attack. In both tables, BERT and RoBERTa show the same statistics since they were pretrained on the same corpus. Notably, KCBERT exhibits a lower rate of unknown tokens in the text than the other two detectors. This gap remains relatively small even when the input text is perturbed. We speculate that this robustness stems from KCBERT's pretraining data, which includes comments posted on online news articles, potentially containing naturally perturbed texts authored by malicious users.

Model	Dataset	Attack Ratio(%)	Text UNK avg	Phoneme UNK avg
BERT	K-HATERS	0	0.4± 1.9	3.5± 5.4
		10	5.3± 5.6	5.3± 6.3
		20	11.8± 9.3	7.5± 7.7
		30	17.9± 12.4	9.8± 9.9
	KoLD	0	0.6± 4.0	4.0± 7.8
		10	5.6± 7.4	5.7± 8.5
		20	13.0± 12.5	8.4± 11.1
		30	19.2± 15.5	10.4± 12.5
RoBERTa	K-HATERS	0	0.4± 1.9	3.5± 5.4
		10	5.3± 5.6	5.3± 6.3
		20	11.8± 9.3	7.5± 7.7
		30	17.9± 12.4	9.8± 9.9
	KoLD	0	0.6± 4.0	4.0± 7.8
		10	5.6± 7.4	5.7± 8.5
		20	13.0± 12.5	8.4± 11.1
		30	19.2± 15.5	10.4± 12.5
KCBERT	K-HATERS	0	0.5± 3.3	1.1± 3.5
		10	1.9± 4.4	1.2± 3.5
		20	3.4± 5.8	1.4± 3.6
		30	4.7± 6.5	1.6± 4.0
	KoLD	0	0.5± 3.7	1.0± 4.3
		10	1.8± 5.2	1.1± 4.4
		20	3.4± 6.9	1.4± 4.7
		30	4.7± 8.7	1.5± 4.9

Table 6: Statistics of unknown tokens in perturbed texts using single-jamo attack and their phoneme sequences

These statistics also offer additional insights into our experimental results. First, the statistics can explain why augmenting phoneme sequences helps mitigate semantic loss caused by phonetic perturbations. When we use a higher attack ratio, the number of unknown tokens increases. So, current models may suffer semantic loss or distortion due to PHISH's phonetic perturbations. By providing phonetic information to the detectors, we could mitigate this loss.

Second, the statistics may explain why KCBERT consistently outperforms the other two detectors. As KCBERT showed fewer unknown tokens, it is highly likely that the model suffers less from semantic loss than the other two models. So, it could achieve higher performance by incorporating semantic and phonetic information, without a considerable loss.

Model	Dataset	Attack Ratio(%)	Text UNK avg	Phoneme UNK avg
BERT	K-HATERS	0	0.4± 1.9	3.5± 5.4
		10	9.9± 6.9	8.1± 7.2
		20	22.9± 11.7	14.7± 10.8
		30	34.2± 15.7	20.7± 13.4
	KoLD	0	0.6± 4.0	4.0± 7.8
		10	10.0± 8.2	8.6± 9.7
		20	24.8± 14.8	15.9± 13.8
		30	37.1± 18.3	22.5± 16.9
RoBERTa	K-HATERS	0	0.4± 1.9	3.5± 5.4
		10	9.9± 6.9	8.1± 7.2
		20	22.9± 11.7	14.7± 10.8
		30	34.2± 15.7	20.7± 13.4
	KoLD	0	0.6± 4.0	4.0± 7.8
		10	10.0± 8.2	8.6± 9.7
		20	24.8± 14.8	15.9± 13.8
		30	37.1± 18.3	22.5± 16.9
KCBERT	K-HATERS	0	0.5± 3.3	1.1± 3.5
		10	6.2± 7.0	1.8± 4.0
		20	12.7± 9.9	2.7± 5.1
		30	18.5± 12.7	3.5± 5.6
	KoLD	0	0.5± 3.7	1.0± 4.3
		10	6.3± 7.9	1.9± 5.7
		20	14.1± 13.3	3.0± 7.3
		30	20.2± 15.8	4.0± 8.6

Table 7: Statistics of unknown tokens in perturbed texts using dual-jamo attack and their phoneme sequences