OpenKorPOS: Democratizing Korean Tokenization with Voting-Based Open Corpus Annotation

Anonymous ACL submission

Abstract

Korean is a language with complex morphology that uses spaces at larger-than-word boundaries, unlike other East-Asian languages. While morpheme-based text generation can provide significant semantic advantages compared to commonly used character-level approaches, Korean morphological analyzers only provide a sequence of morpheme-level tokens, losing information in the tokenization process. Two crucial issues are the loss of spacing information and subcharacter level morpheme normalization, both of which make the tokenization result challenging to reconstruct the original input string, deterring the application to generative tasks. As this problem originates from the conventional scheme used when creating a POS tagging corpus, we propose an improvement to the existing scheme, which makes it friendlier to generative tasks.

On top of that, we suggest a semi-automatic annotation of a corpus by leveraging public analyzers. We vote the surface and POS from the outcome and fill the sequence with the selected morphemes, yielding tokenization with a decent quality that incorporates space information. Our scheme is verified via an evaluation done on an external corpus, and subsequently, is adopted to Korean Wikipedia to construct an open, permissive resource. We compare morphological analyzer performance trained on our corpus with existing methods, then perform an extrinsic evaluation on a downstream task.

1 Introduction

The morphology and script of the Korean language are different from those of Indo-European languages or other East-Asian languages such as Japanese and Chinese (Stratos, 2017; Park et al., 2018). In particular, Korean uses spacing to increase legibility, but not necessarily at word boundaries. The agglutinative properties of Korean result in space tokenized boundaries larger than a word, but smaller than a sentence. This particular unit is called Eojeol, which is a property that is not shared with other languages. Additionally, while there is a well-defined standard for spacing, the rules are complicated.

Prior art suggests that elaborated text processing through morpheme-level analysis is regarded as particularly important in text generation (Kim et al., 2016). In the context of Korean, generally, a single toolkit tends to provide morpheme-level tokenization, morphological analysis and normalization, along with part-of-speech (POS) tagging. For these reasons, the different functions are closely related and commonly trained from a single corpus in an end-to-end manner. This property is inherited from the canonical Sejong Corpus' format (Kim, 2006), which has been exploited to build and train these tools.

However, the standard corpus tagging protocol[1] has seen very few updates since it was initially proposed, and omits crucial information to reconstruct the tokenized results back to their original form, the most obvious being spacing, as suggested in Han et al. (2017). Also, these tokenizers perform stemming and lemmatization following the expected output of the training corpus. For these reasons, research in generation tasks has resorted to using different forms of subword tokenization (Sennrich et al., 2016; Kudo and Richardson, 2018) or work around this limitation by inserting special space tokens as part of the model (Li et al., 2017; Choe et al., 2020). Nonetheless, due to the constraints regarding modification and redistribution[2], there has been no literature addressing this at a large-scale corpus level. We hypothesize that by addressing this from the lowest possible layer, it would allow morpheme-level tokenizers to be more effective

[1]The protocol was designed for morphological analysis, hence did not consider generative tasks, as they were yet to be explored when this was introduced.

[2]The canonical Sejong Corpus (Kim, 2006) is only available to domestic researchers and is distributed under a non-permissive license, which restricts modifications of any form.

	Chinese	Japanese	Korean
Ideographs	Yes	Yes	Rarely
Kana	No	Yes	No
Hangul	No	No	Yes
Spacing	None	None	Optional
Word Order	SVO	SOV	SOV

Table 1: Comparison of CJK languages.

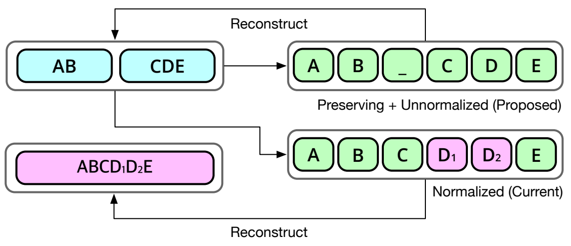


Figure 1: The top (proposed) is with space preservation and no normalization, and the bottom (conventional) is with normalization and no space preservation.

when combined with up-to-date approaches.

We focus on the point that the absence of a large-scale open resource comparable to Sejong hinders the innovation in tokenization research for the Korean language. To apply our method at a corpus level, we leverage outputs from multiple widely-used morpheme-level tokenizers for a voting-based automatic annotation. In specific, we use multiple tokenizers to produce candidate tokenizations, decide the most probable token and morpheme sequence through a voting mechanism, and fill in disputed substring surfaces.

The contribution of this paper is as follows:

- We discuss issues in utilizing tokenization results of current Korean morphological analyzers for generative tasks. We then propose an enhancement to the existing POS tagging protocol to preserve spacing information.
- We note the absence of a universally available large-scale Korean POS tagging corpus under a permissive license. We propose a generation method through semi-automatic annotation and use the output from an ensemble of tokenizers with a voting and filling process.
- We release the POS tagging corpus constructed with our proposed method, under a permissive license open to contributions.

2 Problem Definition

While Korean is commonly classified in the same bucket as Japanese and Chinese, there are details that are commonly misunderstood. Before we define the problems we address in our work, it is important to understand the differences between these three languages, and the problems specific to Korean we would like to address through our work.

2.1 Liberal Whitespaces and Eojeols

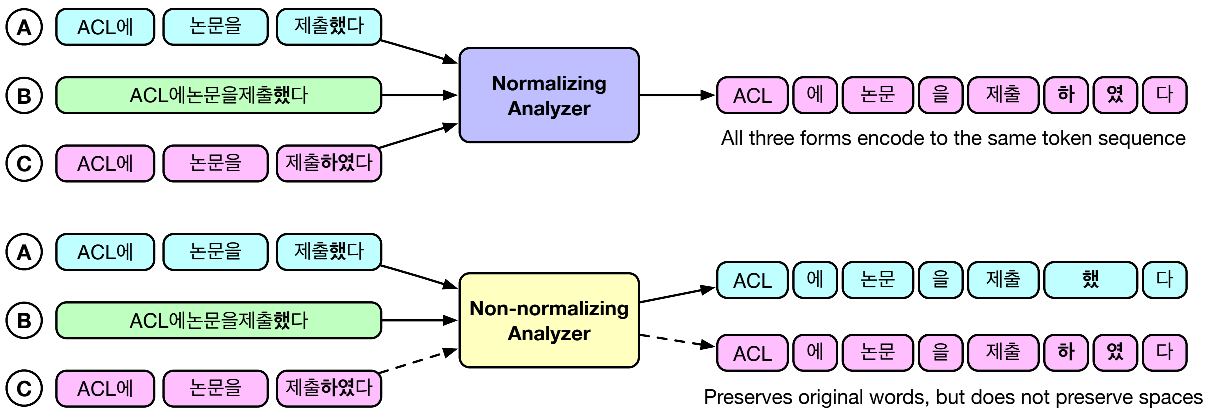
In the comparison Table 1, the most significant difference in the context of tokenization is the usage of whitespaces. What makes Korean different from other languages with spacing is that spacing is indecisive even in formal documents, and often omitted liberally (sometimes entirely) in colloquial text. In particular, Korean spacing is done at the level of an Eojeol, which is a logical block of agglutinated morphemes, that is larger than a word and smaller than a phrase. As described in Figure 1, these morphemes are sometimes not preserved in their original form, that decomposing a sentence and normalizing the morphemes thereof may lead to an output that makes it infeasible to reconstruct back to its original form.

There are multiple cases in the example of Figure 2. In this example, Input A[3] is the most common form of writing. However, Input B, which is the same sentence, but completely stripped of whitespaces is also perfectly legible to the Korean speakers, and is how one may write in a casual context, such as a text message. Input C, is the standard, normalized form that one would find in a formal document or a book - but would be an uncommon form of writing in colloquial contexts.

2.2 Morphological Analyzers as Tokenizers

Before the introduction of subword tokenization, the de-facto method of tokenizing Korean text was to use a library that jointly performs both morphological analysis and POS tagging. The majority of these libraries also perform normalization. In a context where the POS tags are not necessary, the morphs are used as tokens - hence it acts as a tokenizer. This was essential as the number of Eojeol candidates quickly becomes computationally intractable, so breaking it down to morphemes makes it possible to construct a smaller vocabulary.

[3]This sentence means "I submitted a paper to ACL". Specifically, the word boundaries are mainly the functional particles, with the phrase heads at the start of each word:
- ACL.에 / 논문.을 / 제출.했.다
- ACL.to / paper.ACC / submit.PST.DEC
where ACC denotes accusative, PST the past tense, and DEC the declarative.



A
ACL에
논문을
제출했다
B
ACL에논문을제출했다
C
ACL에
논문을
제출하였다
Normalizing Analyzer
ACL 에 논문 을 제출 하 였 다
All three forms encode to the same token sequence
Non-normalizing Analyzer
ACL 에 논문 을 제출 했 다
ACL 에 논문 을 제출 하 였 다
Preserves original words, but does not preserve spaces

Figure 2: Tokenized output compared from morphological analyzers with and without normalization. Line shapes indicate different possible input-output paths.

In traditional NLP methods, this process also helped surfacing stopwords, such as junctions which provides little benefit to task performance, at the same time reducing the amount of verb conjugations by normalization. However, a challenge was inevitable if one implements a text generation model, as the process would not guarantee the information of the original form.

Due to these limitations in currently available morphological analyzers it is impossible to reconstruct the original text. This is demonstrated again in Figure 2, where Input A, B, and C are all tokenized to the exact same output - even if they were originally different. Not only do the majority of analyzers lose information about the original form during normalization, it also does not preserve any information required for reconstructing the original text's spacing. This makes the analyzers unsuitable as a tokenizer for models involving generation tasks, especially if there is normalization involved, as there is no reliable method to reconstruct normalized text back to it's original form.

2.3 Benefits of Morpheme-aware Subwords

Despite the utility of morphological decomposition, many current neural methods use subword tokenization (Sennrich et al., 2016; Kudo and Richardson, 2018) as it allows to construct a robust vocabulary that covers rare or unseen words, while allowing one to set an upper limit on the vocabulary size.

Unfortunately, in the context of a language with a large alphabet as in CJK, this is not always necessarily the case due to the size of the alphabet. With liberal spacing, there is an additional risk of increased complexity training the vocabulary for subword-based algorithms. Prior art such as multilingual BERT (Devlin et al., 2019) work around this by artificially injecting a whitespace at every character f languages. However, it not only unnecessarily increases the sequence length, but also makes it harder for the model learn the linguistic structure as it is effectively operates with a character level vocabulary.

Recent work such as Park et al. (2020) addresses this from a different angle. In their work, they replace the whitespace pre-tokenization (as with BERT) with a morph-level tokenizer, then train a subword tokenizer. They suggest that this improves performance when applied to both transfer learning using a pre-trained language model and machine translation. In their method, to address the lost spacing information we discussed above, they swap whitespaces to a rarely-used Unicode character (U+2583) during encoding, and replace it back to a whitespace when decoded. As we discussed earlier, reconstructing normalized text is not possible with any of the current libraries. For these reasons, they also use a morphological analyzer that does not normalize. In our work, we also restrict the scope to non-normalizing methods.

2.4 Resource Restrictions and Evaluation

In the previous section, we discussed that the utility of morphological analyzer-based tokenization is not limited to lexicon-based methods, but also includes subword-based methods.

However, this comes at a cost - these models have been trained with the Sejong corpus, which is inaccessible to non-Korean citizens. Even if one has access, no modifications (such as correction on errors regarding tokens or POS tags) could be redistributed, so the dataset has effectively been frozen since its initial release in 2006. Another corpus, namely 'Modu Corpus[4]' of NIKL (2020) has the same restrictive license as the Sejong corpus. All these environments make it harder for non-Korean researchers to train a competitive morphological analyzer or POS tagger.

The restrictive nature has contributed to other side effects. Some libraries that have used the corpus to train the analyzers have made local corrections on different subsets of the corpus, which resulted in different training data. On top of this, different or modified subsets of the corpus have been used for evaluation - hence fair comparison between different libraries is currently not possible. We expect this trend to continue unless there is an open training resource for the community to use and improve.

3 Related Work

3.1 Tokenization and Text Generation

Modern neural methods have demonstrated groundbreaking results for generative tasks, (Vaswani et al., 2017; Radford et al., 2018, 2019) all of which rely on tokenized text to be fully reconstructible back to its original form. Korean tasks can also benefit from this, but at the cost of word boundaries not necessarily reflecting the underlying morpheme.

If one needs to tokenize and construct a vocabulary at morpheme-level, utilizing a conventional tokenizer would be the most obvious approach. However, many implementations perform lemmatization and stemming, which is not always reversible. Additionally, information to reconstruct the spacing is often omitted in the tokenized result. Most of all, without spaces, when reconstructed, not only does the text look unnatural, but it also degrades legibility for humans (Cho et al., 2018), bringing further performance degeneration in downstream tasks such as speech synthesis.

For the cases we describe above, an ideal setup lets the tokenizer preserve spacing information while also preserving character-level parity with the original content as much as possible. In our work, we propose an improvement for the POS tagging scheme to guarantee reconstruction. This is implemented as an unofficial extension to the standard POS tag rules defined by the National Institute of Korean Language[5]. We apply this methodology to a redistributable and modifiable corpus, Wikipedia.

[4]https://corpus.korean.go.kr/, distribution has halted from January 18, 2020.

[5]https://www.korean.go.kr/

3.2 Morpheme Tokenizers

In this work, we leverage various existing Korean morpheme analyzing and tokenizing toolkits to annotate a pre-processed, web corpus. In this process, we also incorporate a new POS tag to carry over the original text's spacing. For the point-wise voting mechanism we propose in section 4, we limited tokenizers to those that do not stem or lemmatize. For cases that did normalize, we restricted the choices to tokenizers that provide functionality that allowed us to map a stemmed subcharacter token back to its original character surface.

While all of these implementations are open source[6], the ones that are trained are not reproducible, as to the training data and parameters are not open, and are not quantitatively comparable due to the limitations of the underlying resource used for training as we discussed in the previous section.

3.2.1 Okt tokenizer

Okt[7] is an open-source tokenizer implemented initially with social media posts as its main analysis target. Hence, it performs better than other tokenizers for colloquial Korean sentences. Whether to normalize or stem the sentence is optional, but we used neither here. Unlike other approaches we discuss, Okt is implemented with a very large dictionary combined with dynamic programming methods to search for the ideal tokenization candidate. This model is not a trainable model, and instead is entirely implemented using an algorithmic approach.

3.2.2 MeCab

MeCab (Kudo, 2006) is a widely used, bi-gram Markov model and conditional random field-based (Lafferty et al., 2001) tokenizer originally implemented for Japanese. We use a patched version MeCab for Korean, MeCab-ko[8]. Normalization and stemming is not supported for Korean, and due to this behavior a morph can have multiple POS tags. The open-source model was trained on an undocumented subset of the Sejong corpus, and the standard of quantitative evaluation is absent.

[6]Unfortunately, none of these provide means for citation.

[7]https://github.com/open-korean-text/open-korean-text

[8]https://bitbucket.org/eunjeon/mecab-ko

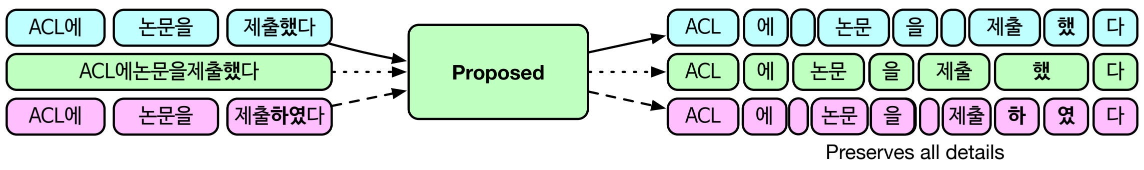


Figure 3: With our proposed changes to the POS tagging protocol, round-trip is guaranteed by preserving everything. Different line shapes indicate different input-output paths.

3.2.3 Khaiii

Khaiii[9] is the first POS tagging toolkit for Korean which uses a deep neural network. It has been inspired by character-level convolutional neural network methods, such as Kim (2014). It processes at a character-level and is implemented as a multi-task model that tokenizes, then predicts the token's POS tag. As Khaiii produces stemmed and lemmatized tokens, we use source surface information to map the token and POS tag back to the original input's substring so that the output format is equal to MeCab[10]. The model was trained on a patched version of the Sejong corpus, which is not publicly available due to the restrictive redistribution license. While there are quantitative performance metrics, as the dataset is closed, it cannot be quantitatively compared with other methods.

4 Proposed Method

We describe *vote* and *fill*, which is a two-fold procedure on how we leverage the conventional tokenizers for corpus generation. Kim et al. (2020) suggested adopting the conventional POS tagged corpora for new annotation, but with rules for erroneous cases, not with multiple tokenizers.

Our method for selecting the ideal candidate token surface was inspired by semantic segmentation tasks such as Ronneberger et al. (2015), a task in the domain of computer vision. It is similar to pointwise label assignment, but while vision tasks operate in a 2D setup, ours is in 1D. We describe how token and POS sequences for a given sentence are decided, and how exceptions are handled. Note that in the overall process, the space information including *space* and *tab* are split as a separate token with blank (SB) as a tag. This information is inherent in voting for both surface and tag.

[9] https://github.com/kakao/khaiii

[10] This is done by matching the source string to the target string at character level, and copying the POS tags from the target string. When the source and target have a mismatching character, we treat that as normalized output and mark the source with the POS tags of each target morph until there is a match between the source and target.

4.1 Vote for surface

For a given sentence, let S_i be the set of surfaces for tokenizer i. An entry $(u, v) \in S_i$ denotes a morpheme (substring) where u is the string index of the first character of the morpheme and v is the index of the last character.

Let S be the set of final surfaces. To select its elements, we first consider the union of S_i, $\bigcup_{i=1}^{M} S_i$, namely the set of all possible surfaces from all tokenizers, $1, ..., M$. For each $(u, v) \in \bigcup_{i=1}^{M} S_i$, we combine weights from the candidate tokenizers. Here, the weight function W is defined:

$$W((u,v)) = \sum_{i=1}^{M} w_i \mathbb{1}_{S_i}((u,v))$$

where w_i is the weight regarding the tokenizer i. We use an indicator function $\mathbb{1}_{S_i}$:

$$\mathbb{1}_{S_i}((u,v)) := \begin{cases} 1 & (u,v) \in S_i \\ 0 & (u,v) \notin S_i \end{cases}$$

To construct S, we order all (u, v) instances in $\bigcup_{i=1}^{M} S_i$ by their weight $W((u, v))$ from the highest to the lowest and assign them into S in order. We do not assign (u, v) in S if it overlaps with pre-assigned surfaces in S. For example, if both $(3, 5)$ and $(4, 5)$ are in the union of S_i and $(3, 5)$ is already assigned in S (due to its weight being higher), then $(4, 5)$ cannot be assigned in S.

4.2 Vote for tag

Let T_i be the POS tag set of tokenizer i. Then,

$$\mathrm{pos}_{(u,v)} \in T_i$$

where $\mathrm{pos}_{(u,v)}$ is the POS tag corresponding to the morpheme regarding the substring (u, v) in S_i. For each (u, v) in S, we gather all possible POS tags

$$T_{(u,v)} = \bigcup_{i=1}^{M} \{\mathrm{pos}^i_{(u,v)} | (u,v) \in S_i\}$$

and calculate weights for each of them, such that:

$$W(\mathrm{pos}_{(u,v)}) = \sum_{i=1}^{M} w_i \mathbb{1}_{T_i}(\mathrm{pos}_{(u,v)})$$

where the indicator function and the weights are defined similarly to the previous section. Choose $p = \mathrm{pos}^*_{(u,v)}$ which yields the maximum weight for $T_{(u,v)}$:

$$\operatorname*{argmax}_p W(p) := \{p | \, p \in T_{(u,v)} \land \\ \forall p' \in T_{(u,v)} : W(p') \leq W(p)\}$$

In ideal cases, we get one $\mathrm{pos}_{(u,v)}$ left for each (u, v) in S. To prevent p from being in tie, appropriate w_i is to be given. Finally, we define T to be the set of the final POS tags corresponding to S.

4.3 Fill

After voting the candidates for surface and tag, we fill the sequence with the resulting S and T. If some (u, v)s are missing in S, we fill the surface (u, v) with POS tag for unknown (UNK). The sentence can be removed if its POS result incorporates a certain amount of UNK. This part is the final step of our algorithm, and thus, may be able to be tackled by utilizing partially annotated data (Sasada et al., 2015) or incomplete annotations (Tsuboi et al., 2008).

To add one of our primary goals, we can detect disputed sentences by checking UNKs in the output. If the tokenization differs a lot due to disagreement, this in turn is expected to increase the frequency of UNKs. This can be used as a metric to identify anomalies, such neologisms not supported by any of the models. However, in the case of using our scheme as real-time voting-based tokenization, UNK may not be desired. In such a case, the user can decide the final tag by choosing a candidate substring (u, v) among $\bigcup_{i=1}^{M} S_i \setminus S$ and its POS that best matches with the corresponding substring (u, v) in terms of exact matching or distance. We found 17,847 sentences (0.44%) containing at least one UNK after this process.

4.4 Corpus Construction

Our goal is to produce a morpheme-level tokenized POS corpus with reconstruction guarantees; for these reasons, we have explicit goals and non-goals. For practical applicability, we constructed the corpus so that the original text can be reconstructed by concatenating the tokens. As a tradeoff, the corpus cannot be used for stemming or lemmatization.

The raw text we used to construct the corpus was collected from a snapshot of the Korean version of Wikipedia[11], which was then pre-processed to remove all Wiki markup, headings, and other metadata. Sentences shorter than a character length threshold t were removed during this clean-up process. With $t = 15$, the process resulted in a total of 4,031,704 usable sentences.

In the annotation process, namely voting and filling, we used the three tokenizers noted in section 2.2. w_i was set to $(1.1, 1.0, 1.0)$, where w_1 was given a higher weight than the others to minimize orphan surfaces. In our experiments, we chose MeCab to have the weight w_1 based on evaluation (2), and uniform weight for the other tokenizers.

5 Experiments

Our scheme yields a morpheme-level, POS tagged corpus of a modest scale. The output of this work can be used for many tasks, such as POS tagging, morpheme level tokenization, language modeling, or small-scale pre-training for transfer learning. The scale of this corpus to other resources is compared in table 4. *Sejong* and *Exobrain*[12] are not openly accessible and nor permissive for modification and redistribution. While *UD Korean* (Chun et al., 2018) and *KLUE-DP* (Park et al., 2021) are accessible, multipurpose resources (e.g., dependency parsing), the size is significantly smaller than that of *Sejong*. Using the corpus we created, first we train a MeCab model with varying sizes of training data sampled from the dataset and compare it with the original MeCab model. Using one of the trained MeCab models, we then perform extrinsic evaluation using a machine translation task and compare it to multiple baselines.

5.1 Morphological Analysis

To first probe if our voting scheme produces better machine annotated data than annotating with a single model, we compare the results using a POS corpus that none of the models have seen. The quantitative analysis was done by comparing our voting scheme with Okt, MeCab, and Khaiii.

We used 683 instances from the *Exobrain*[13] corpus that did not contain any stemmed or lemma-

[11] `ko.wikipedia.org/wiki/`

[12] `http://aiopen.etri.re.kr/`

[13] Adopted since not utilized in any of the baseline training.

	Okt	MeCab	Khaiii	Voted
Surface@Jaccard	0.564	0.825	0.818	**0.848**
POS@Accuracy	0.615	0.944	**0.958**	0.945

Table 2: Our voting scheme compared with other methods. POS accuracy only against matching surfaces.

	10k	15k	30k	50k
Surface@Jaccard	0.802	0.804	0.798	0.799
POS@Accuracy	0.949	0.951	0.952	0.952

Table 3: Comparison with the original MeCab (=1.0). POS accuracy only against matching surfaces.

tized morphemes. With this data, we checked the token and POS match between the ground truth (GT) and the prediction.

Additionally, we sampled 10K sentences from the dataset for human validation, which was then validated and corrected by a linguist. We used the corrected dataset as a gold standard and compared it with the uncorrected samples. This evaluation resulted in a surface score of 0.975 and a POS accuracy of 0.992. The modest results demonstrate that our scheme can produce a reasonably accurate dataset.

The performance is measured in two ways. First, we use a modified Jaccard index to measure the reliability of tokenization. Originally, the Jaccard index for a sentence is defined as the proportion of common surfaces among the union of GT and predicted surfaces. However, to ensure that the tokenization and tagging are correctly evaluated when tokens are repeated in a sentence, we attach the order of appearance to each token so as to distinguish the overlapped morphemes, which may possibly have different POS tags. The final Jaccard index is averaged over all sentences in the test corpus. Second, the accuracy of predicted POS tags is calculated using common surfaces between GT and the predictions. We observed that voting produced more reliable results than using a single model, as can be seen in Table 2.

Additionally, to verify that our data can be used to train a morphological analyzer, we used the data to train a MeCab model and compared this to the original MeCab. The model trained with a small data [14] reproduced around 80% of the performance that the original model has, as seen in Table 3.

As the training protocol has not been officially documented, we used default parameters for training. We hypothesize that if training is done with the same training parameters and data size as the original model used, the gap can be further reduced. We observed that our model splits words much more aggressively, which contributed to mismatched surfaces.

[14]Less than 1.5% of the entire data.

	Eojeols	Purpose	Open
Sejong	10,066,722	POS Tagging	×
Modu Corpus	3,006,660	POS Tagging	×
Exobrain	33,131	Universal	×
UD Korean	532,598	Universal	o
KLUE-DP	136,987	Universal	o
Ours	55,154,053	POS Tagging	o

Table 4: Comparison of scale with known corpora. *Open* indicates open access with a permissive license.

Level	BPE	MeCab	Khaii	Khaii-N	Ours
Morph	28.88	**36.73**	35.18	30.26	36.03
Eojeol	12.52	**17.68**	15.70	12.82	17.21

Table 5: BLEU score comparison of different tokenization schemes. Khaii-N is Khaiii with normalization.

5.2 Machine Translation

For extrinsic evaluation, we used Marian NMT (Junczys-Dowmunt, 2019) trained to perform English to Korean (en-ko) translation. The tokenization and evaluation protocol followed the work in Park et al. (2020). We used the news data from the AI Hub machine translation dataset[15], which consists of approximately 800K English-Korean sentence pairs. For our experiments, we used 40K sentences for test and validation and the remainder for training. The translation model used is an RNN-based encoder-decoder model with attention, using a shared 85K subword-level vocabulary trained with byte-pair encoding (BPE) after morpheme-level tokenization (Sennrich et al., 2017), trained for 10 epochs.

As our work focuses on improving generation performance, we limited our evaluation to en-ko since it adequately displays the tendency of reconstruction regarding tokenization. The 10K model from our previous experiment was used as a pre-tokenizer for BPE and compared against BPE without pre-tokenization, and three other models as the pre-tokenizer. Due to the limitations of MeCab[16] which was used as the probe model for our cor-

[15]Though the evaluation with accessible benchmarks such as Park et al. (2016) is recommended for reproducibility, we could not adopt those in training and test due to various quality issues such as mistranslations and typos.

[16]As MeCab uses whitespaces as breaks, to use our corpus significant modifications were needed.

pus, spacing was emulated through a special token (U+2583). This allows reversible reconstruction, as seen in Figure 3.

We compared the different approaches using BLEU at morpheme-level following WAT2019 (Nakazawa et al., 2019) and Eojeol-level. For morpheme-level evaluation, the final detokenized output was re-tokenized with MeCab. Due to Korean's agglutinative nature, Eojeol-level is an incredibly difficult task, primarily when evaluated with BLEU. Agglutinations of certain morphs such as junctions are often optional, and this can negatively affect the BLEU score even when the predicted output is perfectly coherent. On top of that, we evaluate if the model performs spacing perfectly, which is a difficult task even for a native speaker. As can be seen from the results in Table 5, while the performance of the original Mecab models is slightly better,our model trained only on a small subset of data is better than that of other tokenization schemes in a translation context.

6 Discussion

6.1 Why Our Scheme and Corpus?

Our primary aim is to create an open and redistributable corpus that can be utilized in model training with further refinement. The vote and fill scheme achieve these goals, given that the resulting corpus shows adequate performance when evaluated on usual tasks. However, to ensure quality, human annotation is required.

One clear merit of our tagging scheme is that the conventional corpus designers can obtain a reliable POS tagged draft for any raw corpus s/he adopts. It is common practice to refine a machine annotated corpus with human annotation, and MeCab is often used to perform this kind of machine annotation in practice. However, the machine output is usually not sufficient as a draft due to domain-specific OOV issues. Our scheme helps the training process leverage other candidate tokenizers with the voting-based decision.

The other advantage of our resulting corpus is that it delivers an open, accessible resource that allows future refinement and extension. As Wikipedia content is distributed under a share-alike license, further redistribution mandates the same license policy. This includes our work, but derivatives of it as well, effectively making this an open source project. We assume this can encourage other community members to engage in the analysis and enhancement of the proposed resource.

6.2 Limitation

Normalization Though our approach suggests an incremental enhancement of tokenization and POS tagging from the status quo, we do not handle the normalization of lexicons in our process. Thus, for further usage of stemmed or lemmatized tokens, the users may necessitate additional post-processing or a module which specializes in this task. Normalization is related to but is a different issue from tokenization; thus, we leave it as a separate work in our study.

Lack of library support While our scheme is interoperable with existing tools, we noticed an oversight during our experiments. The probe tokenizer we used (MeCab) breaks at spaces, resulting in this information being lost during training. In the experiments, we emulated spacing by replacing it with a special character, but existing libraries will require modifications to use the proposed scheme. Alternatively, a novel tokenization method that incorporates this could also be potential future work.

Quality of tokenization We acknowledge our approach's limitation and that the result is not fully at the quality level of a human-annotated gold standard. This prevents our corpus from being adopted as a benchmark dataset. However, the human validation results suggest that our dataset is capable of producing a dataset of modest quality, and with incremental error corrections we believe it would be possible to establish the subset of our corpus as a benchmark.

7 Conclusion

In this work, we identify a constraint in the standard protocol of creating Korean POS tagging corpora, namely that the construction does not account for the necessity of spacing. We demonstrate that such limitation of the corpus propagates to the tokenizers trained with those, limiting the applicability to a generative task.

We then propose a novel, voting-based method for this at the corpus generation level, creating an unprecedented large-scale open resource with this mitigation applied to enable universal access to a Korean POS tagging and morpheme level tokenization research. Unlike previous datasets, ours can be incrementally enhanced by the greater research community.

Ethical Considerations

We provide the dataset that is automatically annotated by the publicly available POS tagging/tokenization modules. The raw corpus is Korean Wikipedia, which is available under Creative Commons Attribution-ShareAlike 3.0 Unported License. Some of the datasets used for the comparison are restricted to non-Korean researchers and are referred to claim the exclusiveness of current open resources. The MT corpus used in the evaluation is free and accessible with a simple sign-in. Still, it is considered difficult to attain for non-Korean researchers, and the redistribution is restricted. The usage was inevitable due to the lack of a usable open machine translation corpus.

Our data construction and experiment do not involve the human subject and manual works. The corpus constructed in this paper is based on a widely-referred but not yet POS annotated dataset. Wikipedia is also known as a community-contributed document set that is refined with public discussions.

The proposed data regards POS tagging and tokenization, which is more syntactic and might not involve bias or hate issues. However, due to the vast size of the corpus, we could not yet guarantee there exists the automatic inferences that may induce any form of harm. As the resource is malleable through community contributions, we hope to react and remove problematic data as they are discovered quickly.

References

Won Ik Cho, Sung Jun Cheon, Woo Hyun Kang, Ji Won Kim, and Nam Soo Kim. 2018. Real-time automatic word segmentation for user-generated text. *arXiv preprint arXiv:1810.13113*.

Byeongseo Choe, Ig-hoon Lee, and Sang-goo Lee. 2020. Korean morphological analyzer for neologism and spacing error based on sequence-to-sequence. *Journal of KIISE*, 47(1):70–77.

Jayeol Chun, Na-Rae Han, Jena D Hwang, and Jinho D Choi. 2018. Building universal dependency treebanks in korean. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Gyeong-Eun Han, Seul-Ye Baek, and Jae-Soo Lim. 2017. Open sourced and collaborative method to fix errors of sejong morphologically annotated corpora. In *Annual Conference on Human and Language Technology*, pages 228–232. Human and Language Technology.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.

Hansaem Kim. 2006. Korean national corpus in the 21st century sejong project. In *Proceedings of the 13th NIJL International Symposium*, pages 49–54. National Institute for Japanese Language Tokyo.

Taeyoung Kim, Pum Mo Ryu, Hansaem Kim, and Hyo-Jung Oh. 2020. Unified methodology of multiple pos taggers for large-scale korean linguistic gs set construction. *Journal of KIISE*, 47(6):596–602.

Yang-hoon Kim, Yong-keun Hwang, Tae-gwan Kang, and Kyo-min Jung. 2016. LSTM language model based Korean sentence generation. *The Journal of Korean Institute of Communications and Information Sciences*, 41(5):592–601.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. pages 1746–1751.

Taku Kudo. 2006. Mecab: Yet another part-of-speech and morphological analyzer. *http://mecab. sourceforge. jp*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando C N Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*.

Jianri Li, EuiHyeon Lee, and Jong-Hyeok Lee. 2017. Sequence-to-sequence based morphological analysis and part-of-speech tagging for korean language with convolutional features. *Journal of KIISE*, 44(1):57–62.

Toshiaki Nakazawa, Nobushige Doi, Shohei Higashiyama, Chenchen Ding, Raj Dabre, Hideya Mino, Isao Goto, Win Pa Pa, Anoop Kunchukuttan, Yusuke

Oda, Shantipriya Parida, Ondřej Bojar, and Sadao Kurohashi. 2019. Overview of the 6th workshop on Asian translation. In *Proceedings of the 6th Workshop on Asian Translation*, pages 1–35, Hong Kong, China. Association for Computational Linguistics.

National Institute of Korean Languages. 2020. NIKL CORPORA 2020 (v.1.0).

Jungyeul Park, Jeen-Pyo Hong, and Jeong-Won Cha. 2016. Korean language resources for everyone. In *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Oral Papers*, pages 49–58, Seoul, South Korea.

Kyubyong Park, Joohong Lee, Seongbo Jang, and Dawoon Jung. 2020. An empirical study of tokenization strategies for various Korean NLP tasks. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 133–142, Suzhou, China. Association for Computational Linguistics.

Sungjoon Park, Jeongmin Byun, Sion Baek, Yongseok Cho, and Alice Oh. 2018. Subword-level word vector representations for korean. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2429–2438.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook 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, Alice Oh, Jungwoo Ha, and Kyunghyun Cho. 2021. Klue: Korean language understanding evaluation.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*.

Tetsuro Sasada, Shinsuke Mori, Tatsuya Kawahara, and Yoko Yamakata. 2015. Named entity recognizer trainable from partially annotated data. In *Conference of the Pacific Association for Computational Linguistics*, pages 148–160. Springer.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. 2017. Nematus: a toolkit for neural machine translation.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Karl Stratos. 2017. A sub-character architecture for korean language processing. *arXiv preprint arXiv:1707.06341*.

Yuta Tsuboi, Hisashi Kashima, Shinsuke Mori, Hiroki Oda, and Yuji Matsumoto. 2008. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 897–904, Manchester, UK. Coling 2008 Organizing Committee.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

A Appendices

A.1 Environment: Corpus Construction

Corpus construction was done by parallelizing the tokenization work across 128-threads. For these experiments, we used a Dual AMD EPYC 7551 server with 256GBs of RAM. The entire process took approximately 72 hours. No co-processors were used for this process.

A.2 Environment: Machine Translation

Machine translation was done on a Dual Intel Xeon Gold 6148 server with 360GBs of RAM, parallelized across four Nvidia Tesla V100-SXM2 (16GB) GPUs. Each of the eight experiments took approximately 5 hours, resulting in about 40 hours of wall-clock time. Additionally, a grand total of 88 hours were used to search for adequate training parameters and architectures.

A.3 Environment: Others

Other experiments, such as MeCab evaluation and training, were done on the authors' laptops and desktops, so we do not consider the computation budget used here significant enough for disclosure.

A.4 Training Parameters

The Marian parameters used to train the en-ko translation model are as follows:

- -w 12500
- –max-length 100
- –mini-batch-fit
- –mini-batch 1000
- –maxi-batch 1000
- –beam-size 12
- –normalize=1
- –valid-mini-batch 64
- –early-stopping 5
- –after-epochs 10
- –cost-type=ce-mean-words
- –enc-type bidirectional
- –enc-depth 1
- –enc-cell-depth 4
- –dec-depth 1
- –dec-cell-base-depth 8
- –dec-cell-high-depth 1
- –tied-embeddings-all
- –layer-normalization
- –dropout-rnn 0.1
- –label-smoothing 0.1
- –learn-rate 0.0003
- –lr-decay-inv-sqrt 16000
- –optimizer-params 0.9 0.98 1e-09
- –clip-norm 5
- –sync-sgd
- –exponential-smoothing