Analysis of the Penn Korean Universal Dependency Treebank (PKT-UD): Manual Revision to Build Robust Parsing Model in Korean

Tae Hwan Oh[♠], Ji Yoon Han[♠], Hyonsu Choe[♠], Seokwon Park[♠], Han He,[◇] Jinho D. Choi[◇], Na-Rae Han[♣], Jena D. Hwang[♡], Hansaem Kim[♠]

[♠]Yonsei University, Seoul, South Korea
[◇]Emory University, Atlanta GA 30322, USA
[♣]University of Pittsburgh, Pittsburgh PA 15260, USA
[♡]Allen Institute For Artificial Intelligence, Seattle WA 98103, USA

{ghksl0604,clinamen35,choehyonsu,pswon27}@yonsei.ac.kr, han.he@emory.edu
jinho.choi@emory.edu, naraehan@pitt.edu, jenah@allenai.org, khss@yonsei.ac.kr

Abstract

In this paper, we first open on important issues regarding the Penn Korean Universal Treebank (PKT-UD) and address these issues by revising the entire corpus manually with the aim of producing cleaner UD annotations that are more faithful to Korean grammar. For compatibility to the rest of UD corpora, we follow the UDv2 guidelines, and extensively revise the part-of-speech tags and the dependency relations to reflect morphological features and flexible word-order aspects in Korean. The original and the revised versions of PKT-UD are experimented with transformer-based parsing models using biaffine attention. The parsing model trained on the revised corpus shows a significant improvement of 3.0% in labeled attachment score over the model trained on the previous corpus. Our error analysis demonstrates that this revision allows the parsing model to learn relations more robustly, reducing several critical errors that used to be made by the previous model.

1 Introduction

In 2018, Chun et al. (2018) published on three dependency treebanks in Korean that followed the latest guidelines from the Universal Dependencies (UD) project, that was UDv2. These treebanks were automatically derived from the existing treebanks, the Penn Korean Treebank (PKT; Han et al. 2001), the Google UD Treebank (McDonald et al., 2013), and the KAIST Treebank (Choi et al., 1994), using head-finding rules and heuristics.

This paper first addresses the known issues in the original Penn Korean UD Treebank, henceforth PKT-UD v2018, through a sampling-based analysis (Section 3), and then describes the revised guidelines for both part-of-speech tags and dependency relations to handle those issues (Section 4). Then, a transformer-based dependency parsing approach using biaffine attention is introduced (Section 5) to experiment on both PKT-UD v2018 and the revised version, henceforth PKT-UD v2020 (Section 6). Our analysis shows a significantly reduced number of mispredicted labels by the parsing model developed on PKT-UD v2020 compared to the one developed on PKT-UD v2018, confirming the benefit of this revision in parsing performance. The contributions of this work are as follows:

1. Issue checking in PKT-UD v2018.
2. Revised annotation guidelines for Korean and the release of the new corpus, PKT-UD v2020.
3. Development of a robust dependency parsing model using the latest transformer encoder.

2 Related Works

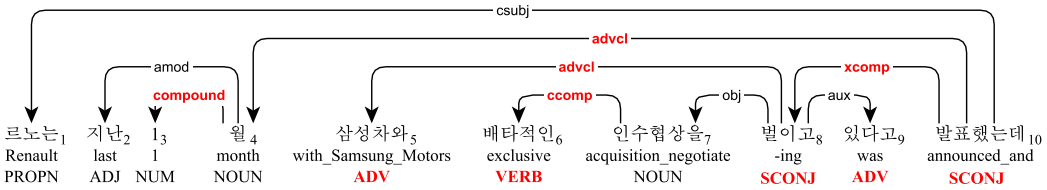
2.1 Korean UD Corpora

According to the UD project website,[1] three Korean treebanks are officially registered and released: the Google Korean UD Treebank (McDonald et al., 2013), the Kaist UD Treebank (Choi et al., 1994), and the Parallel Universal Dependencies Treebank (Zeman et al., 2017). These treebanks were created by converting and modifying the previously existing treebanks. The Korean portion of the Google UD Treebank had been re-tokenized into the morpheme level in accordance with other Korean corpora, and systematically corrected for several errors (Chun et al., 2018). The Kaist Korean UD Treebank was derived by automatic conversion using head-finding rules and linguistic heuristics (Chun et al., 2018). The Parallel Universal Dependencies Treebank was designed for the CoNLL 2017 shared task on Multilingual Parsing, consisting of 1K sentences extracted from newswires and Wikipedia articles.

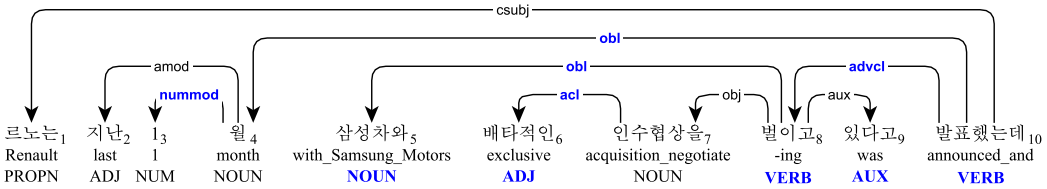
The Penn Korean UD Treebank and the Sejong UD Treebank were registered on the UD website

[1] https://universaldependencies.org

Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task, pages 121–130
Virtual Meeting, July 9, 2020.
https://doi.org/10.18653/v1/P17



(a) Example from v2018 where the labels in revision are indicated by the red bold font.



(b) Example from v2020 where the revised labels are indicated by the blue bold font.

Figure 1: Example from v2018 and v2020, that translates to "*Renault announced last January that it was negotiating an exclusive acquisition with Samsung Motors, and ...*". This example continues in Figure 2.

as well but unreleased due to their license issues. Similar to the Kaist UD Treebank, the Penn Korean UD Treebank[2] was automatically converted into UD structures from phrase structure trees (Chun et al., 2018). The Sejong UD Treebank was also automatically converted from the Sejong Corpus, a phrase structure Treebank consisting of 60K sentences from 6 genres (Choi and Palmer, 2011).

Treebank	GKT	KTB	PUD	PKT	Sejong
Sentences	6k	27k	1k	5k	60k
Tokens	80k	350k	16k	132k	825k
Released	O	O	O	X	X
Unit	Eojeol	Eojeol	Eojeol	Eojeol	Eojeol
Genre	Blog, News	Litr, News, Acdm, Mscr	Blog, News	News	Litr, News, Acdm, Mscr

Table 1: Korean UD Treebanks. Each abbreviations indicate genres of source texts: webblogs(Blog), newswire(News), literatures(Litr), academic(Acdm), manuscripts(Mscr).

In a related effort, the Electronic and Telecommunication Research Institute (ETRI) in Korea conducted a research on standardizing dependency relations and structures (Lim et al., 2015). This effort resulted in the establishment of standard annotation guidelines of Korean dependencies, giving rise to various related efforts that focused on the establishment of Korean UD guidelines that better represent the unique Korean linguistic features. These studies include Park et al. (2018) who focused on the mapping between the UD part-of-speech (POS) tags and the POS tags in the Sejong Treebank, and Lee et al. (2019) and Oh (2019) who provided in-depth discussions of applicability and relevance of UD's dependency relation to Korean.

2.2 Penn Korean UD Treebank

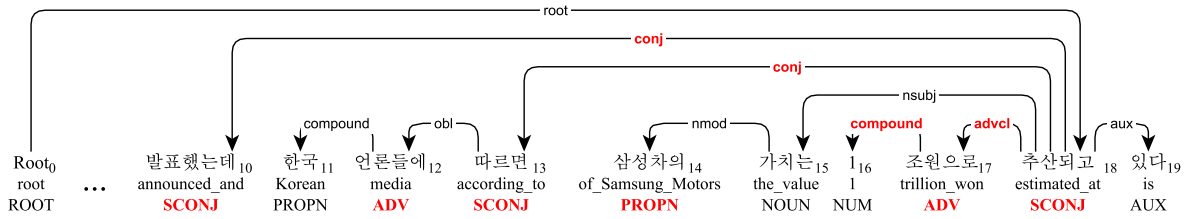
As mentioned in Section 2.1, the Penn Korean UD Treebank (PKT-UD v2018) was automatically derived from phrase-structure based the Penn Korean Treebank and the results were published by Chun et al. (2018). Even so, it currently does not number among the Korean UD treebanks officially released corpora under the UD project website.

Our effort to officially release Chun et al. (2018)'s PKT-UD v2018 has uncovered numerous mechanical errors caused by the automatic conversion and few other unaddressed issues, leading us to a full revision of this corpus. PKT-UD v2018 made targeted attempts at addressing a number of language-specific issues regarding complex structures such as empty categories, coordination structures, and allocation of POS tags with respect to dependency relations. However, the efforts were limited, leaving other issues such as handling of copulas, proper allocation of verbs according to their verbal endings, and grammaticalized multi-word expressions were unanswered. Thus, this paper aims to address those remaining issues while revising PKT-UD v2018 to clearly represent phenomena in Korean.

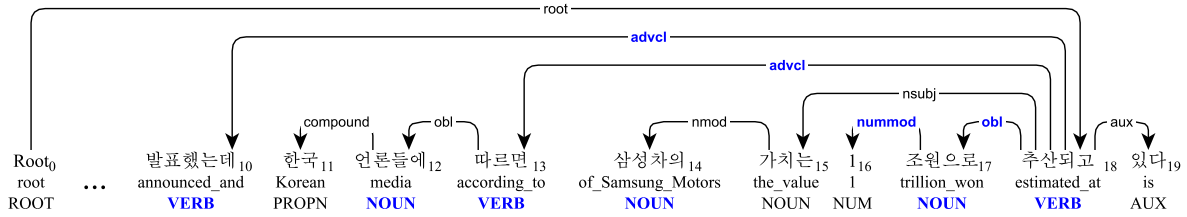
3 Observations in PKT-UD v2018

The Penn Korean Treebank (PKT) was originally published as a phrase-structure based treebank by Han et al. (2001). PKT consists of 5,010 sentences

[2]The annotation with the word-forms of the Penn Korean UD Treebank can be found here: https://github.com/emorynlp/ud-korean.



(a) Example from v2018 where the labels in revision are indicated by the red bold font.



(b) Example from v2020 where the revised labels are indicated by the blue bold font.

Figure 2: Continuing example from Figure 1 that translates to "*... announced ..., and according to Korean media, the value of Samsung Motors is estimated at 1 trillion won*".

from Korean newswire including 132,041 tokens.[3] Following the UDv2 guidelines, Chun et al. (2018) systematically converted PKT to PKT-UD v2018. While this effort achieved a measure of success at providing phrase-structure-to-dependency conversion in a manner consistent across three different treebanks with distinct grammatical frameworks, it stopped short of addressing more nuanced issues that arise from aligning grammatical features of Korean, that is a heavily agglutinative language, to the universal standards put forth by UDv2. In building PKT-UD v2018, the POS tags were largely mapped in a categorical manner from the Penn Korean POS tagset. The dependency relations on the other hand were established via head-finding rules that relied on Penn Korean Treebank's existing function tags, phrasal tags, and morphemes.

Chun et al. (2018) did make a few targeted attempts at teasing apart more fine-grained nuances of grammatical functions. For example, the PKT POS tag (XPOS) `DAN` was subdivided into the UD POS tag (UPOS) `DET` for demonstrative prenominals (e.g., 이 (*this*), 그 (*the*), and the UPOS `ADJ` for attributive adjectives (e.g., 새 (*new*), 헌 (*old*)) in the recognition that the XPOS `DAN`, focusing primarily on grammatical distribution, conflated two semantically distinct elements. However, such efforts were limited in scope, and the project did not examine the full breadth of language-specific issues.

Moreover, the converted annotation was found to contain a share of mechanical errors. A case in point, what should have been 5,010 sentences were found to contain 5,036 roots, suggesting low-level parsing errors. Additionally, a manual examination of the first five sentences in the corpus uncovered a variety of syntactic errors that raised an alarm. The worst of the five examined sentences is shown in Figure 1 (and continued in Figure 2) with errors in both the UPOS and the dependency relation labels (DEPREL). While we will not delve into particulars of each error seen in this example, the example provides a general sense for the extent of errors existent that merited our attention.

These observed issues inspired us to revise PKT-UD v2018, with the aim of producing cleaner syntactic annotations that would be more faithful to the Korean grammar. The following section provides specifics of the revision content.

4 PKT-UD Revisions

4.1 UPOS Revision

Revision of the UPOS portion of the resource was done from the ground up. That is, instead of correcting PKT-UD v2018's UPOS annotations, we implemented a new mapping from XPOS to UPOS after a careful re-examination of the original mapping schema. In particular, we consulted the POS mapping guidelines by Park et al. (2018) whose morphological tagset, carried over from the Sejong Project (Kim, 2006), differs from PKT's in some key aspects. However, we found their nuanced view of grammatical characteristics and typology of Korean in reference to the UDv2 very much applicable.

[3] While most Korean resources have what is known as *Eojeol* representing a token and white space is used as delimiter, PTK tokenizes apart symbols, punctuation and even occasional morphemes where strictly required by syntactic structure.

The followings illustrate key ideas of of our UPOS revision approach. Below and throughout this paper, we italicize XPOS labels (e.g., *DAN*) so they are visually distinct from UPOS labels (e.g., ADJ).

Copulas mapped to ADJ One major target of revision was the scope of the UPOS adjective label ADJ in Korean, which includes typical predicative adjectives such as '예쁘-' (*pretty*) and '다르-' (*different*). As mentioned in Section 3, PKT-UD v2018 already extended the ADJ label to include the closed class of adjectives whose distribution is limited to pre-nominal, attributive use which had been grouped together with the determiner category *DAN* in the original PKT. In our current work, we further extend the ADJ label to encompass the copula: *CO* ('-이-' (*be*)). In Korean, '-이-' (*be*) is a copula particle that attaches to a nominal to produce a predicate, much like the English '*be*'. However, such copula-derived predicates in Korean are known to share semantic and syntactic traits with adjectives rather than verbs, chief among which being their inability to take on the present/habitual aspect verbal ending '-는다' (*do*) which is only allowed on verbs. In light of this, we made a decision to map all instances of XPOS' *CO* to UPOS' ADJ.

Consistent NOUN focusing on morpheme roles Korean is well-known as an agglutinative language, and *Josa*s (postpositions) are extremely common nominal suffixes that can indicate a variety of syntactic roles of the whole Eojeol unit (Figure 3). For example, when an adverbial case particle ('에', *PAD*) attaches to a noun, the resulting Eojeol serves the syntactic role of an adverb. When a conjunctive particle ('와', *PCJ*) is used, the Eojeol functions as a noun conjunct. Consequently, PKT-UD v2018 mapped ADV to the former and CCONJ to the latter.

학교	학교+**에**	학교+**와**
hakkyo	*hakkyo*+***PAD***	*hakkyo*+***PCJ***
(*school*)	(***at*** *school*)	(*school* ***and***)

Figure 3: Korean postpositions, marked in bold.

However, this distinction underscores a syntactic role rather than a morphological one: while the syntactic role changes with the attachment of the postposition, the POS of the noun itself remains unaffected. UPOS, as a marker that solely demonstrates morphological characteristics of Eojeol rather than its syntactic function, should reflect the morphological status of the nominal. Therefore, we made a decision to allocate the NOUN label to these cases.

Verbal endings signal VERB Korean has verbal endings on predicates that dictate the syntactic role of Eojeol (Figure 4). In PKT-UD v2018, predicates marked with *ENM* (nominalization verbal ending) and *ECS* (conjunctive ending) are mapped to NOUN and SCONJ, respectively. However, as with the earlier case involving nominals, these verbal ending suffixes should not be treated as fundamentally altering the underlying POS of the predicate itself. This work revises both cases of UPOS to VERB. Extending the same principle, parallel cases with the same verbal endings involving an adjective or a copula were likewise re-assigned to ADJ.

먹+**다**	먹+**기**	먹+**고**
mek+***ta***	*mek*+***ki***	*mek*+***ko***
(*Eat*)	(*Eat**ing***)	(*Eat* ***and***)

Figure 4: Korean verbal endings, marked in bold.

Statistics of v2018 and v2020 The complete distributions of PKT-UD v2018 and v2020 are listed in Table 2.

UPOS	v2018	v2020	PC
ADJ	3,431	7,034	105.0 ↑
ADP	1,251	1,425	13.9 ↑
ADV	15,174	2,851	81.2 ↓
AUX	2,263	4,060	79.4 ↑
CCONJ	2,453	377	84.6 ↓
DET	685	685	0.0
NOUN	46,866	58,367	24.5 ↑
NUM	7,931	7,602	4.1 ↓
PART	464	290	37.5 ↓
PRON	857	1,142	33.3 ↑
PROPN	12,257	12,769	4.2 ↑
PUNCT	13,428	13,428	0.0
SCONJ	9,780	533	94.6 ↓
SYM	376	376	0.0
VERB	13,855	21,102	52.3 ↑
X	970	0	100.0 ↓
Total	132,041	132,041	0.0

Table 2: Universal POS tagset comparison between the 2018 and 2020 versions of the Penn Universal Dependency Treebank. v2018/v2020: the number of tokens in those versions respectively, PC: percentage change.

4.2 DEPREL Revision

In re-examining PTK-UD v2018's dependency relations, we consulted two existing dependency annotation guidelines for Korean: Lee et al. (2019)

and Oh (2019). They offer a thorough analysis on applicability of the universal dependency relation labels to Korean, and further identify a list of dependency relations such as `iobj`, `xcomp`, `expl`, and `cop` (among others) as not suited for capturing characteristics of Korean grammar. Additionally, where applicable, we took into consideration the UD Japanese Treebank (Asahara et al., 2018), since Japanese exhibits many parallel syntactic phenomena as another strictly head-final agglutinative language (Kanayama et al., 2018).

Reevaluation of `iobj` We turned our attention to `iobj`, the DEPREL label for indirect object. We found PKT-UD v2018's decision to assign nominals with dative case markings to `iobj` questionable, for the following reasons. First, unlike English, where word order distinguishes indirect objects from direct objects (e.g. "She gave me:`iobj` a box:`obj`"), Korean has no such structural constraint that forms the basis for identifying instances of `iobj`. The only potential identifier, then, is dative postpositions such as '-에게'(*to*) and '-한테'(*by*), which correspond roughly to English preposition '*to*' as in "She gave it ***to*** me". The problem is, these markers do not exclusively encode the dative case, as seen in examples such as "개**에게** 물렸다" ("*I was bit* ***by*** *a dog*").

Hence, we adopted a new approach of reassigning all instances of `iobj` to the oblique relation `obl`. This move brings language-internal consistency, as postpositions, in many instances, can simply be dropped if contextually recoverable, rendering any such nominals practically indistinguishable from other nominal adverbials that are assigned to `obl`. This overall approach is also in line with UD Japanese Treebank, where `iobj` is categorically absent and 'に (*ni*)', a postposition whose usage largely parallels the two Korean postpositions above, mapping to `obl`.

Standardizing verbal predicates As shown in Figure 5, Korean predicates take on various syntactic functions depending on the attached verbal ending. Predicates with the declarative verbal ending '-다' (*ta*) are assigned to `root`, which is straightforward. Endings '-은' (*un*) and '-을' (*ul*) on the other hand turn the verb into a modifier to an upcoming noun; the `acl` relation therefore is the best fit here. Predicates with endings such as '-어서' (*ese*) and '-게' (*key*) modify other predicates, which calls for an `advcl` assignment. In PKT-UD v2018, these cases had received an array of inconsistent allocations such as clausal complements (`ccomp`/`xcomp`), auxiliaries (`aux`), and conjuncts (`conj`). These were corrected to `acl` and `advcl`.

먹+다	먹+은/을	먹+어서
mek-ta	*mek-un/ul*	*mek-ese*
(*eat*)	(*ate/to-eat*)	(*eat because*)

Figure 5: Examples of Korean verbal ending.

Orphaned postpositions and verbal endings In Korean, verbal endings and postpositions are bound to verbs and nominals, respectively, and cannot occupy their own Eojeol. In natural text, however, they can occasionally be separated from the constituent they attach to via quotation marks, white spaces, or parentheses. PKT-UD v2018 had assigned such orphaned bound morphemes to UPOS of `PART` (particle) and `ADP` (adposition) with the DEPREL of `mark` (marker) and `case` (case marker), respectively as seen in Figure 6.

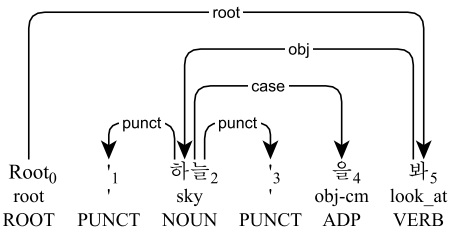


Figure 6: PKT-UD v2018 treatment of separated postposition '-을' (*ul*) in "'하늘'을 봐 (Look at the 'sky')".

However, verbal endings and postpositions can express syntactic function only if they are attached to their modifying predicates and nominals. While PKT-UD v2018's assignment of the UPOS and DEPREL are not categorically incorrect, they address morphological relationship between these morphemes rather than their syntactic relationship. That is, even if these bound morphemes are notationally distanced from their heads by punctuation or white spaces, they form a single syntactic unit with their nominals and postpositions. Hence, `mark` and `case` were updated to `goeswith`, used for divided words as seen in Figure 7, making it clear that the seemingly separate Eojeols (e.g. nominal and postposition) are actually one unit.

Orphaned copulas Similar revisions were applied to copulas. Korean copula morpheme '-이-' (*i*) combines with a nominal on the left and a verbal

ending to the right. These copulas too can occasionally be detached via intervening punctuation or white space. To such cases, PKT-UD v2018 had assigned `cop` as the DEPREL. These instances have been updated to `goeswith` in accordance with the treatment given to verbal endings and postpostions.

`roots` and `flats` The number of `root` is adjusted from 5,036 to 5,010 after correcting sentences with zero or more roots. Additionally, DEPREL of Eojeols that used to be incorrectly mapped to `compound` are now assigned to `flat`.

Statistics of v2018 and v2020 The complete DEPREL distributions of PKT-UD v2018 and v2020 are listed in Table 3.

DEPREL	v2018	v2020	PC
acl	1,488	11,210	653.4 ↑
advcl	11,636	5,086	56.3 ↓
advmod	2,964	3,125	5.4 ↑
amod	1,595	1,593	0.1 ↓
appos	1,182	1,173	0.8 ↓
aux	4,807	4,061	15.5 ↓
case	1,548	0	100.0 ↓
ccomp	9,858	1,989	79.8 ↓
cc	785	473	39.7 ↓
compound	28,908	21,433	25.9 ↓
conj	9,960	7,155	28.2 ↓
cop	418	0	100.0 ↓
csubj	8,014	8,012	0.0 ↓
dep	609	10	98.4 ↓
det	685	685	0.0
fixed	528	589	11.6 ↑
flat	18	739	4,005.6 ↑
goeswith	0	2,199	100.0 ↑
iobj	222	0	100.0 ↓
mark	1,003	0	100.0 ↓
nmod	5,555	5,501	1.0 ↓
nsubj	4,012	4,114	2.5 ↑
nummod	154	7,341	4,666.9 ↑
obj	9,823	9,849	0.3 ↑
obl	3,357	16,891	403.2 ↑
orphan	0	9	100.0 ↑
punct	13,073	13,794	5.5 ↑
root	5,036	5,010	0.5 ↓
xcomp	4,803	0	100.0 ↓
Total	132,041	132,041	0.0

Table 3: Universal dependency label comparison between v2018 and v2020 of the Penn Universal Dependency Treebank. v2018/v2020: the number of tokens in those versions respectively, PC: percentage change.

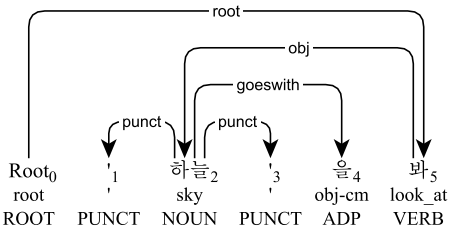


Figure 7: Revision of the DEPREL of the separated postposition 을 at "'하늘'을 봐 (Look at the 'sky')" in PKT-UD v2020, where `case` relation for orphaned postposition revised to `goeswith`.

5 Parsing Approach

Our dependency parsing model is based on the biaffine parser using contextualized embeddings such as BERT (Devlin et al., 2019) that has shown the state-of-the-art results on both syntactic and semantic dependency parsing tasks in multiple languages (He and Choi, 2020). This model is simplified from the original biaffine parser introduced by Dozat and Manning (2017) such that trainable token embeddings are removed and lemmas are used instead of word forms. This section proposes an even more simplified model that no longer uses embeddings from POS tags, so it can be easily adapted to languages that do not have dedicated POS taggers, and drops the Bidirectional LSTM encoder while integrating the transformer layers directly into the biaffine decoder so that it minimizes the redundancy of having multiple encoders for the generation of contextualized embeddings.

Given an input sentence, every token w_i is first segmented into one or more sub-tokens by the SentencePiece tokenizer (Kudo and Richardson, 2018) and fed into a transformer. The output embedding corresponding to the first sub-token of w_i is treated as the embedding representation of w_i, say $\mathbf{e}_i$, and fed into four types of multilayer perceptron (MLP) layers to extract features for w_i being a head (*-h) or a dependent (*-d) for the arc relations (arc-*) and the labels (rel-*) (k and l are the dimensions of the arc and label representations, respectively):

$$\mathbf{h}_i^{(\text{arc-h})} = \text{MLP}^{(\text{arc-h})}(\mathbf{e}_i) \in \mathbb{R}^{k\times 1}$$
$$\mathbf{h}_i^{(\text{arc-d})} = \text{MLP}^{(\text{arc-d})}(\mathbf{e}_i) \in \mathbb{R}^{k\times 1}$$
$$\mathbf{h}_i^{(\text{rel-h})} = \text{MLP}^{(\text{rel-h})}(\mathbf{e}_i) \in \mathbb{R}^{l\times 1}$$
$$\mathbf{h}_i^{(\text{rel-d})} = \text{MLP}^{(\text{rel-d})}(\mathbf{e}_i) \in \mathbb{R}^{l\times 1}$$

All feature vectors, $\mathbf{h}_1^*, \ldots, \mathbf{h}_n^*$, from each representation are stacked into a matrix (n is the number

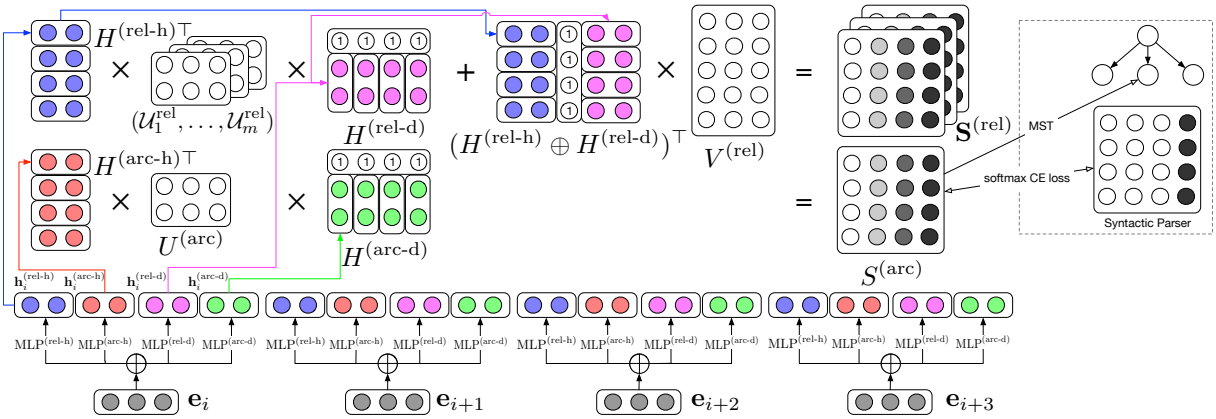


Figure 8: The overview of our transformer-based biaffine dependency parsing model.

of tokens in a sentence); these matrices together are used to predict dependency relations among every token pairs. Note that bias terms are appended to the feature vectors $\mathbf{h}_i^{(*\text{-d})}$ that represent dependent nodes to estimate the likelihood of a certain relation given only the head node:

$$
\begin{aligned}
H^{(\text{arc-h})} &= (\mathbf{h}_1^{(\text{arc-h})}, \ldots, \mathbf{h}_n^{(\text{arc-h})}) \in \mathbb{R}^{k \times n} \\
H^{(\text{arc-d})} &= (\mathbf{h}_1^{(\text{arc-d})}, \ldots, \mathbf{h}_n^{(\text{arc-d})}) \oplus \mathbf{1} \in \mathbb{R}^{(k+1) \times n} \\
H^{(\text{rel-h})} &= (\mathbf{h}_1^{(\text{rel-h})}, \ldots, \mathbf{h}_n^{(\text{rel-h})}) \in \mathbb{R}^{l \times n} \\
H^{(\text{rel-d})} &= (\mathbf{h}_1^{(\text{rel-d})}, \ldots, \mathbf{h}_n^{(\text{rel-d})}) \oplus \mathbf{1} \in \mathbb{R}^{(l+1) \times n}
\end{aligned}
$$

The bilinear and biaffine classifiers are then used for the arc and label predictions respectively, where $U^{(\text{arc})}$, $U_i^{(\text{rel})}$ and $V^{(\text{rel})}$ are trainable parameters, and m is the number of dependency labels. In particular, a separate weight matrix $U_i^{(\text{rel})}$ is dedicated for the prediction of each label:

$$
\begin{aligned}
S^{(\text{arc})} &= H^{(\text{arc-h})\top} \cdot U^{(\text{arc})} \cdot H^{(\text{arc-d})} \in \mathbb{R}^{n \times n} \\
\mathcal{U}_i^{(\text{rel})} &= H^{(\text{rel-h})\top} \cdot U_i^{(\text{rel})} \cdot H^{(\text{rel-d})} \in \mathbb{R}^{n \times n} \\
\mathbf{S}^{(\text{rel})} &= (\mathcal{U}_1^{(\text{rel})}, \ldots, \mathcal{U}_m^{(\text{rel})}) \\
&\quad + (H^{(\text{rel-h})} \oplus H^{(\text{rel-d})})^\top \cdot V^{(\text{rel})} \in \mathbb{R}^{m \times n \times n}
\end{aligned}
$$

Once the arc score matrix $S^{(\text{arc})}$ and the label score tensor $\mathbf{S}^{(\text{rel})}$ are generated by those classifiers, the Chu-Liu-Edmond's maximum spanning tree (MST) algorithm is applied to $S^{(\text{arc})}$ for the arc prediction, then the label with largest score in $\mathbf{S}^{(\text{rel})}$ corresponding to the arc is taken for the label prediction:

$$
\begin{aligned}
arc &= \text{MST}(S^{(\text{arc})}) \\
label &= \text{argmax}(\mathbf{S}^{(\text{rel})}[\text{index}(arc)])
\end{aligned}
$$

6 Experiments

To extrinsically assess the quality of our revision, parsing models are separately developed on PKT-UD v2018 and v2020; in other words, v2018 models are trained and evaluated on PKT-UD v2018 whereas v2020 models are trained and evaluated on PKT-UD v2020. The transformer-based parsing approach in Section 5 is used to develop all models. For each version of the corpus, three models are developed by initializing neural weights with different random seeds and the average accuracy and its standard deviation is reported for each version. The entire corpus is divided into the training (TRN), development (DEV), and evaluation (TST) sets by following the 80/10/10% split (Table 4).

	TRN	DEV	TST
# of Sentences	4,010	501	500
# of Tokens	105,947	13,088	13,023

Table 4: Statistics of the data split.

The multilingual BERT[4] is used as the transformer encoder in our parsing models (Devlin et al., 2019). All models are optimized by the sum of softmax cross-entropy losses on the gold dependency heads and labels. AdamW (Loshchilov and Hutter, 2019) is used as the optimizer with the learning rate of 5e-06 for the BERT weights and 5e-05 for the rest. The learning rate is scheduled as a combination of both linear warm-up and decay phases. The models are trained for 100 epochs with a batch size of 150. Following the standard practice, we evaluate our best models with the unicode punctuation ignored using the unlabeled attachment score (UAS) and the labeled attachment score (LAS).

[4] https://github.com/google-research/bert/blob/master/multilingual.md

Table 5 shows the results achieved by the v2018 and v2020 models. The v2020 model shows a significantly improvement of 3.0% in LAS over the v2018 model. This makes sense because the major parts of the revision are dedicated to DEPREL consistency, yielding more robust parsing performance in labeling. The v2020 model also gives a good improvement of 0.6% in finding dependency arcs. The improved parsing results ensure the higher quality annotation in PKT-UD v2020 that is encouraging.

	UAS	LAS
v2018	90.7 (±0.2)	86.0 (±0.1)
v2020	**91.3** (±0.1)	**89.0** (±0.1)

Table 5: Results by the v2018 and v2020 models.

7 Error analysis

PKT-UD v2018 We perform an error analysis on the parsing outputs generated by the v2018 model. Our analysis shows that the head error occurred in 1,360 Eojeols and the label error occurred in 4,292 Eojeols. Table 6 shows the distribution of head and label errors per label based on the revised test set. The relations `advcl`, `nummod`, `acl`, and `obl` have a high error rate, which are due to the inconsistencies seen in the data we handled by establishing clear criteria. Moreover, the labels `goeswith` and `flat` saw 100% error, again, due to the errors we observed during the revision process.

DEPREL	Error	Percentage
obl	1,294	30.15%
acl	961	22.39%
nummod	777	18.1%
advcl	462	10.76%
goeswith	203	4.73%
conj	99	2.31%
compound	96	2.24%
flat	91	2.12%
ccomp	77	1.79%
etc	232	5.41%
Total	4,292	100%

Table 6: DEPREL error of PKT-UD v2018.

There is an observable trend in these errors. For example, a number of error cases report `advcl` as `xcomp`, `conj`, or `ccomp` while `nummod` tends to be wrongly parsed to `compound`, `acl` to `ccomp`, and `obl` to `advcl`. Multiple cases of parsing errors due to errors in the UPOS are also found. Incorrect UPOS appears to commit errors while allocating edge and DEPREL. The annotation guideline based on XPOS is already described in Section 4.

PKT-UD v2020 After revising the data according to the criteria presented in Section 4, many improvements have been made. The error rate of `advcl` decreased from 98.93% to 2.36%, the `nummod` also decreased significantly from 97.37% to 0.5%, and the `acl` error from 86.73% to 0.9%. The error rate of `obl` was also reduced from 79.14% to 5.5%. In addition, the error rate is reduced for `goeswith` and `flat`. In the case of `ccomp`, errors decreased by more than 35% from 44.51% to 8.67%. These results is indicative of the effect of improving training data by ensuring consistency of annotations.

DEPREL	v2018	v2020
obl	1,294	90
acl	961	10
nummod	777	4
advcl	462	11
goeswith	203	3
conj	99	85
compound	96	83
flat	91	34
ccomp	77	15
etc	232	134
Total	4,292	469

Table 7: DEPREL error comparison between PKT-UD v2018 and v2020.

8 Conclusion

In this study, we revise the Penn Korean Universal Dependency Treebank (PKT-UD) and compare parsing performance between models trained on the original and revised versions of PKT. Our new guidelines follow the UDv2 guidelines. UPOS and DEPREL are revised to reflect Korean morphological features and flexible word-order aspects with reference to Korean UD studies such as Park et al. (2018), Lee et al. (2019), and Oh (2019). In UPOS, `ADJ`, `NOUN`, and `VERB` are revised extensively. In DEPREL, `iobj`, `acl`, `advcl`, and `goeswith` are thoroughly revised. The revision results showing the percentage change of each label are presented in Table 2 and Table 3.

As a result of the parsing experiment, the v2020 model improves UAS by 0.6% and LAS by 3.0% over the v2018 model. In particular, `obl`, `acl`,

nummod, and advcl errors are significantly reduced. This study, which improves parsing accuracy by applying characteristics of Korean, can also contribute to improve the quality of other Korean UD treebanks. In the future, we will explore the possibility of extending PKT-UD with enhanced dependency types[5] by incorporating empty categories from the original PKT.

References

Masayuki Asahara, Hiroshi Kanayama, Takaaki Tanaka, Yusuke Miyao, Sumire Uematsu, Shinsuke Mori, Yuji Matsumoto, Mai Omura, and Yugo Murawaki. 2018. Universal dependencies version 2 for japanese. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jinho D. Choi and Martha Palmer. 2011. Statistical dependency parsing in korean: From corpus generation to automatic parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 1–11. Association for Computational Linguistics.

Key-Sun Choi, Young S Han, Young G Han, and Oh W Kwon. 1994. Kaist tree bank project for korean: Present and future development. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pages 7–14. Citeseer.

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.

Timothy Dozat and Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR'17.

Chung-hye Han, Na-Rae Han, Eon-Suk Ko, Martha Palmer, and Heejong Yi. 2001. Penn korean treebank: Development and evaluation. In *Proceedings of the 16th Pacific Asia Conference on Language, Information and Computation*, pages 69–78.

[5] https://universaldependencies.org/u/overview/enhanced-syntax.html

Han He and Jinho D. Choi. 2020. Establishing strong baselines for the new decade: Sequence tagging, syntactic and semantic parsing with bert. In *Proceedings of the 33rd International Florida Artificial Intelligence Research Society Conference*, FLAIRS'20.

Hiroshi Kanayama, Na-Rae Han, Masayuki Asahara, Jena D Hwang, Yusuke Miyao, Jinho D Choi, and Yuji Matsumoto. 2018. Coordinate structures in universal dependencies for head-final languages. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 75–84.

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.

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.

Chanyoung Lee, Taehwan Oh, and Hansaem Kim. 2019. A study on universal dependency annotation for korean. *Language Fact and Perspectives*, 47(0):1–11.

Joon-Ho Lim, Yongjin Bae, Hyunki Kim, Yunjeong Kim, and Kyu-Chul Lee. 2015. Korean Dependency Guidelines for Dependency Parsing and Exo-Brain Language Analysis Corpus. In *Proceedings of the 27tht Annual Conference on Human and Cognitive Language Technology*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97.

Taehwan Oh. 2019. Study on universal dependencies for korean. Master's thesis, Yonsei University, Seoul, Korea.

Hyejin Park, Taehwan Oh, and Hansaem Kim. 2018. Universal pos tagset for korean. *The Korean Society for Language and Information*, 22(3):67–89.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka

Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.