How do Transformer-Architecture Models Address Polysemy of Korean Adverbial Postpositions?

Seongmin Simon Mun, Guillaume Desagulier

How do Transformer-Architecture Models Address Polysemy of Korean Adverbial Postpositions?

Seongmin Mun
Chosun University, Republic of Korea
seongmin.mun@chosun.ac.kr

Guillaume Desagulier
Paris VIII University & UMR 7114, MoDyCo, France
guillaume.desagulier@univ-paris8.fr

Abstract

Postpositions, which are characterized as multiple form-function associations and thus polysemous, pose a challenge to automatic identification of their usage. Several studies have used contextualized word-embedding models to reveal the functions of Korean postpositions. Despite the superior classification performance of previous studies, the particular reason how these models resolve the polysemy of Korean postpositions is not enough clear. To add more interpretation, for this reason, we devised a classification model by employing two transformer-architecture models—BERT and GPT-2—and introduces a computational simulation that interactively demonstrates how these transformer-architecture models simulate human interpretation of word-level polysemy involving Korean adverbial postpositions *-ey*, *-eyse*, and *-(u)lo*. Results reveal that (i) the BERT model performs better than the GPT-2 model to classify the intended function of postpositions, (ii) there is an inverse relationship between the classification performance and the number of functions that each postposition manifests, (iii) model performance is affected by the corpus size of each function, (iv) the models' performance gradually improves as the epoch proceeds, and (vi) the models are affected by the scarcity of input and/or semantic closeness between the items.

1 Introduction

Polysemy, one type of ambiguity, occurs when one form delivers multiple, and yet related, meanings/functions(Glynn and Robinson, 2014). Traditional word-embedding models have shown an unsatisfactory level of performance in polysemy interpretation(e.g., Bae et al., 2014, 2015; Kim and Ock, 2016; Lee et al., 2015; Mun and Shin, 2020; Shin et al., 2005). This is mainly due to the technical nature of these models: they are *static* in that a single vector is assigned to each word (Desagulier, 2019; Ethayarajh, 2019; Liu et al., 2019a). To overcome this issue, recent studies have proposed a *contextualized* word-embedding model which considers neighborhood information about a polysemous word on the basis of sequences of words around the target word (Klafka and Ettinger, 2020; Loureiro and Jorge, 2019; Michalopoulos et al., 2021). These models have achieved a good level of performance in many natural language processing tasks (Devlin et al., 2018; Radford et al., 2018a; Liu et al., 2019b; Radford et al., 2018b; Lan et al., 2019). Among these models, transformer-architecture models such as Bidirectional Encoder Representations from Transformer (BERT; Devlin et al., 2018) and Generative Pre-Training 2 (GPT-2; Radford et al., 2018b) shows the best performance for this task of polysemy interpretation (Haber and Poesio, 2021; Soler and Apidianaki, 2021; Yenicelik et al., 2020).

Despite a good deal of research on transformer-architecture models in English, very few studies have investigated transformer-architecture-based polysemy interpretation in languages that are typologically different from English. We therefore attend to Korean, a Subject-Object-Verb language with overt case-marking through a postposition—a bound morpheme which adds grammatical functions to a content word where it is attached (Sohn, 1999). A Korean postposition normally involves many-to-many associations between form and function. As such a postposition is polysemous (Choo and Kwak, 2008). For example, an adverbial postposition *-(u)lo* (*-ulo* after a consonant) is interpreted as six major functions: criterion (CRT), direction (DIR), effector (EFF), final state (FNS), instrument (INS), and location (LOC) (Shin, 2008). For instance, the following sentence involving the postposition *-(u)lo* as a marker of INS (instrument) as in (1).

(1) 전선이 연줄로
censen-i yencwul-lo
wire-NOM connection.wire-INS

Proceedings of Deep Learning Inside Out (DeeLIO 2022):
The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 11 - 21

-ey		*-eyse*		*-(u)lo*	
Function	Frequency	Function	Frequency	Function	Frequency
LOC	1,780	LOC	4,206	FNS	1,681
CRT	1,516	SRC	647	DIR	1,449
THM	448			INS	739
GOL	441			CRT	593
FNS	216			LOC	158
EFF	198			EFF	88
INS	69				
AGT	47				
Total	4,715	Total	4,853	Total	4,708

Table 1: By-function frequency list of *-ey*, *-eyse*, and *-(u)lo* in cross-validated corpus

Note. Abbreviation: AGT = agent; CRT = criterion; DIR = direction; EFF = effector; FNS = final state; GOL = goal; INS = instrument; LOC = location; SRC = source; THM = theme

감겼다.
kam-ki-ess-ta.
wind-PSV-PST-DECL
'The wire wound around with the connection wire.'

Several studies have used transformer-architecture models to address the word-level polysemy of Korean adverbial postpositions (e.g., Bae et al., 2020a,b; Hong et al., 2020; Mun, 2021) with the model performance (measured through a *F*-score) ranging from 0.776 (Park et al., 2019) to 0.856 (Bae et al., 2020a). Notably, the particular reason for the transformer architecture's superior performance over the others is somewhat unclear (Puccetti et al., 2021; Yun et al., 2021). To add more interpretation of the model performance, we devised a classification model by employing BERT and GPT-2. In order to further understand how transformer-architecture models recognize word-level polysemy, we propose a transformer-architecture-based visualization system for polysemy interpretation of three adverbial postpositions, *-ey*, *-eyse*, and *-(u)lo*, which are frequently documented in the previous studies (e.g., Cho and Kim, 1996; Jeong, 2010; Nam, 1993; Park, 1999; Song, 2014).

2 Methods

2.1 Creating the Input Corpus

The Sejong primary corpus (Sejong corpus is available at: https://www.korean.go.kr), the representative corpus in Korean, does not code the information about the functions of postpositions directly in each sentence (which is necessary for model training). We thus annotated a portion of the original corpus data manually. For this purpose, we extracted sentences involving only one postposition and predicate. We did this treatment to control for additional confounding factors which might have interfered with the performance of our model. We then extracted 5,000 sentences randomly for each postposition from the initial dataset.

Three native speakers of Korean annotated each postposition for its function in this 15,000-sentence corpus. Fleiss' kappa scores were 0.948 (*-ey*), 0.928 (*-eyse*), and 0.947 (*-(u)lo*), which are considered

Index	Label	Sentence
1,862	1	[CLS] 한참 만에 오반장이 침묵을 깼다. [SEP]
1,863	1	[CLS] 정말 오랫만에 먹어보는 고기였다. [SEP]
1,864	1	[CLS] 옛날 구한말에 유명한 얘기가 있었죠? [SEP]
1,865	1	[CLS] 한밤중에 신나게 한바탕했지요. [SEP]
1,866	1	[CLS] 그런데 몇 시에 왔어? [SEP]
1,867	1	[CLS] 겨울에 꽃이라니요. [SEP]
1,868	1	[CLS] 아침에 엄마한테 돈을 달랬어요. [SEP]
1,869	1	[CLS] 결혼은 반드시 적령기에 해야 한다. [SEP]
1,870	1	[CLS] 한 달에 얼마씩은 정확하게 들어오니까. [SEP]
1,871	1	[CLS] 그럼 일 주일 후에 뵙겠습니다. [SEP]

Index	Label	Sentence
1,862	1	한참 만에 오반장이 침묵을 깼다.
1,863	1	정말 오랫만에 먹어보는 고기였다.
1,864	1	옛날 구한말에 유명한 얘기가 있었죠?
1,865	1	한밤중에 신나게 한바탕했지요.
1,866	1	그런데 몇 시에 왔어?
1,867	1	겨울에 꽃이라니요.
1,868	1	아침에 엄마한테 돈을 달랬어요.
1,869	1	결혼은 반드시 적령기에 해야 한다.
1,870	1	한 달에 얼마씩은 정확하게 들어오니까.
1,871	1	그럼 일 주일 후에 뵙겠습니다.

Figure 1: Example sentences used in the training for BERT (left) and GPT-2 (Right)

'almost perfect' according to the Kappa scale. We further excluded instances which showed disagreement among the human annotators. The final corpus consisted of 4,715 sentences for *-ey*, 4,853 sentences for *-eyse*, and 4,708 sentences for *-(u)lo*. Table 1 presents the detailed by-function frequency list of the three postpositions[1].

2.2 Creating Training and Test Sets

In order to make the training and test sets, we made three separate columns: index (the unique number of each sentence), label (the intended function of each postposition in each sentence), and sentence. For the sentence of BERT, we added [CLS] ('classification'; indicating the start of a sentence) before a sentence and [SEP] ('separation'; indicating the end of a sentence) after a sentence to indicate where the sentence starts and ends (Figure 1 left); no such addition occurred in GPT-2. We then split the corpus into two sub-sets, one with 90 per cent of the corpus for the training and with the remaining 10 per cent of the corpus for the testing.

2.3 Developing BERT and GPT-2 Models

For the BERT model training, we transformed the input data into three embedding types—*token* embedding, *position* embedding, and *segment* embedding (c.f., Devlin et al., 2018)—in the following ways. First, for the *token* embedding, we used *KoBertTokenizer* for the sentence tokenization; the maximum number of tokens for each sentence was set to 128. Second, for the *position* embedding, we converted each token into numeric values indicating unique indices of the tokens in the vocabulary of KoBERT. Third, for the *segment* embedding, we converted the number of tokens of each sentence into 128 numeric values using 0 (i.e., not existed) or 1 (i.e., existed). The labels of the data indicating the intended function of each postposition in the sentence were stored separately.

After the input creation, we set the parameters related to both of model training such as *batch size* (16), *epoch* (50), *seed* (42), *sequence length* (128), *epsilon* (0.00000008), and *learning rate* (0.00002), as advised by previous studies (e.g., McCormick, 2019; Vázquez et al., 2020; Wu et al., 2019). We then employed BERT and GPT-2 pre-trained language models in order to obtain high performance of outcomes: KoBERT (Jeon et al., 2019) for BERT and KoGPT-2-base-v2 (Jeon et al., 2021) for GPT-2. The BERT model training proceeded as follows. First, we loaded KoBERT through the function *BertForSequenceClassification* from *transformers* (Wolf et al., 2019). Second, we fine-tuned the pre-trained model by using the training set, with a view to reducing loss values and updating the *learning rate* for better classification performance of the model. Third, we loaded the testing set to evaluate whether the fine-tuned model successfully recognized the intended functions of each postposition in each sentence. In this part, the rates of F-score for each function and the total F-score rate (i.e., F-score) were calculated by comparing the intended function of each postposition in each test sentence against the parsed version returned by the model. Lastly, we employed t-distributed Stochastic Neighbor Embedding (t-SNE; Maaten and Hinton, 2008) for dimension reduction of classification embeddings from the postposition by each epoch. In addition, to statistically confirm the changes of sentence-level embedding outcomes by each epoch, we performed density-based clustering (Sander et al., 1998). These outcomes were fed into the visualization system [2].

The input treatment process and the training process of GPT-2 are almost the same as the BERT training process. For the input treatment, first, the BERT model used symbols to mark the start and the end of each input, but no such addition occurred in GPT-2 training. Second, BERT uses wordpiece algorithm for the *token* embedding (Sennrich et al., 2016), but GPT-2 uses byte pair encoding algorithm (Gage, 1994). For the training process, first, BERT operates on the basis of masked language modeling and next-sentence prediction for generating a pre-trained model, whereas GPT-2 uses general language modeling by using a huge size corpus. Second, the BERT model conducts learning in bi-direction, while GPT-2 conducts learning in uni-direction. In addition, GPT-2 loaded KoGPT2 through the function *GPT2ForSequenceClassification* and *PreTrainedTokenizerFast* from *transformers* (Wolf et al., 2019).

2.4 Developing the Visualization System

In order to better understand how BERT and GPT-2 recognize the word-level polysemy, we developed

[1] Our corpus is available at: https://github.com/seongmin-mun/Corpora/tree/main/APIK

[2] Code can be found at: https://github.com/seongmin-mun/Visualization/tree/master/2022/PostTransformers

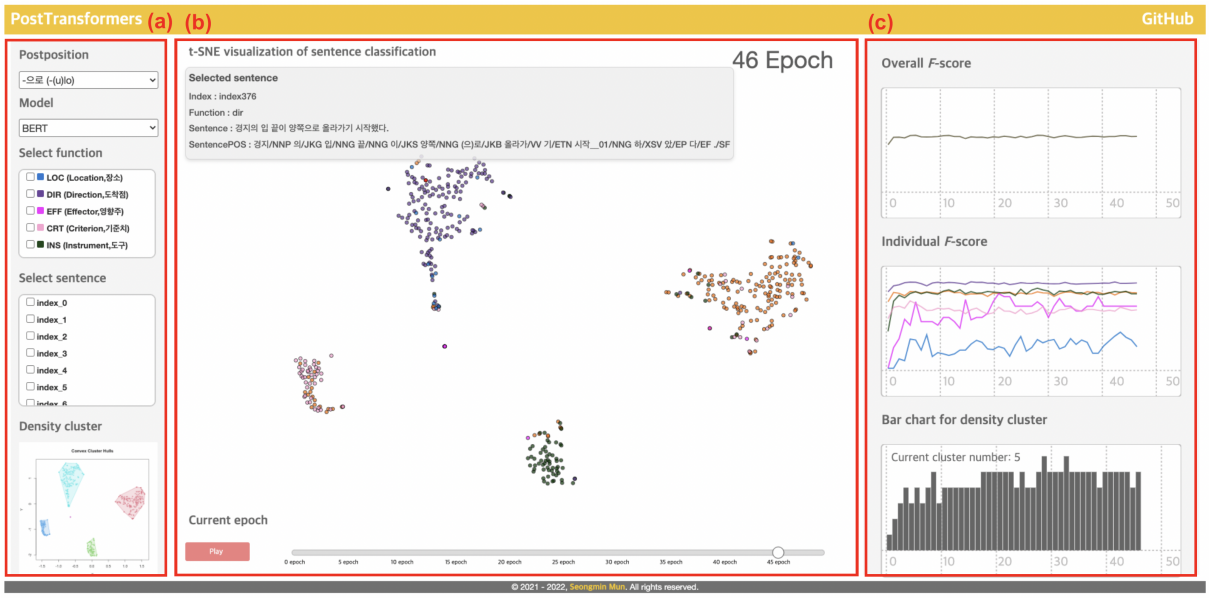

Figure 2: The overview interface of the visualization system

a visualization system by using the test set under the two-dimensional distribution. For the system interface, we created three areas for the demonstration of model performance: a distributional map for sentence-level embeddings, *F*-score charts relating to the model, and graphs for the density-based clustering[3].

2.4.1 Distributional Map for Sentence-Level Embeddings

The distributional map as in Figure 2b presents the relationship between the sentences with the selected postposition (represented as dots) involving different functions (represented as colors). A slider at the bottom of the map allows for changing the epochs; the patterns of clustering change as the slider moves. Each dot shows the details of the sentence (e.g., an index of the selected sentence, the intended function used in the sentence, the original sentence) once the mouse pointer is located on the dot. For the manipulating of visualization outcomes, Figure 2a provides options to select the checkboxes to highlight and tracking interesting sentences according to the function of these postpositions, the models, and the index number.

2.4.2 *F*-score Charts

The right side of the system as in Figure 2c provides users with various information about the model performance: overall *F*-score and by-function *F*-score in the classification task by epoch. This section also provides *F*-score rates of each function by hovering around the mouse pointer onto the specific-colored lines.

2.4.3 Graphs for the Density-Based Clustering

The bar chart at the bottom right side of the system presents the number of clusters produced by the

[3]The visualization system available at: https://seongmin-mun.github.io/Visualization/2022/PostTransformers/index.html

Epoch	Classification performance (*F*-score)								
	Overall	*AGT*	*CRT*	*EFF*	*FNS*	*GOL*	*INS*	*LOC*	*THM*
1	0.677	0	0.812	0.286	0	0.125	0	0.802	0.472
10	0.739	0.286	0.84	0.37	0.5	0.444	0.222	0.813	0.646
20	0.758	0.25	0.848	0.514	0.474	0.478	0.286	0.827	0.705
30	0.745	0.25	0.823	0.571	0.542	0.448	0.375	0.816	0.688
40	0.73	0.222	0.805	0.537	0.522	0.478	0.3	0.82	0.66
50	0.747	0.25	0.839	0.529	0.5	0.413	0.353	0.817	0.705
Average	0.744	0.217	0.837	0.531	0.499	0.435	0.29	0.823	0.651

Table 2: By-function *F*-score for the BERT model: *-ey*

Epoch	Classification performance (*F*-score)		
	Overall	*LOC*	*SRC*
1	0.893	0.94	0.509
10	0.88	0.933	0.431
20	0.874	0.93	0.358
30	0.87	0.927	0.376
40	0.878	0.931	0.468
50	0.87	0.928	0.364
Average	0.875	0.93	0.373

Table 3: By-function *F*-score for the BERT model: *-eyse*

Epoch	Classification performance (*F*-score)						
	Overall	*CRT*	*DIR*	*EFF*	*FNS*	*INS*	*LOC*
1	0.681	0.532	0.82	0	0.714	0.396	0
10	0.799	0.644	0.924	0.462	0.805	0.794	0.167
20	0.794	0.595	0.915	0.714	0.807	0.818	0.2
30	0.799	0.598	0.908	0.545	0.808	0.829	0.296
40	0.792	0.612	0.915	0.667	0.797	0.794	0.24
50	0.803	0.627	0.915	0.667	0.812	0.809	0.286
Average	0.795	0.626	0.911	0.604	0.805	0.803	0.233

Table 4: By-function *F*-score for the BERT model: *-(u)lo*

Epoch	Classification performance (*F*-score)								
	Overall	*AGT*	*CRT*	*EFF*	*FNS*	*GOL*	*INS*	*LOC*	*THM*
1	0.514	0	0.579	0	0	0	0	0.617	0
10	0.7	0	0.8	0.5	0.148	0.31	0	0.794	0.591
20	0.675	0	0.793	0.39	0.235	0.222	0	0.768	0.56
30	0.672	0	0.784	0.364	0.421	0.328	0.2	0.771	0.495
40	0.687	0	0.811	0.324	0.41	0.25	0	0.768	0.592
50	0.685	0	0.814	0.333	0.333	0.254	0	0.768	0.582
Average	0.68	0.003	0.796	0.386	0.335	0.24	0.061	0.769	0.546

Table 5: By-function *F*-score for the GPT-2 model: *-ey*

Epoch	Classification performance (*F*-score)		
	Overall	*LOC*	*SRC*
1	0.857	0.923	0
10	0.851	0.918	0.217
20	0.864	0.925	0.214
30	0.849	0.915	0.305
40	0.843	0.912	0.296
50	0.851	0.917	0.28
Average	0.844	0.912	0.272

Table 6: By-function *F*-score for the GPT-2 model: *-eyse*

model. This chart also provides a hovering function, providing the actual number of clusters per epoch. The particular hovering activity is interlocked with the density cluster view, located at the bottom left of the system, by presenting the clustering results according to the selected epoch.

Epoch	Classification performance (*F*-score)						
	Overall	*CRT*	*DIR*	*EFF*	*FNS*	*INS*	*LOC*
1	0.473	0.03	0.549	0	0.575	0.099	0
10	0.675	0.611	0.765	0.471	0.701	0.583	0.207
20	0.664	0.619	0.782	0.429	0.658	0.568	0.273
30	0.696	0.621	0.801	0.5	0.721	0.587	0.222
40	0.683	0.585	0.799	0.462	0.691	0.612	0.24
50	0.694	0.603	0.803	0.5	0.702	0.635	0.222
Average	0.676	0.588	0.782	0.425	0.69	0.591	0.226

Table 7: By-function *F*-score for the GPT-2 model: *-(u)lo*

3 Results: Four Case Studies

In order to report the transformer-architecture models' performance of classifying the functions of postpositions and assess how our visualization system works, we conducted four case studies.

3.1 Which Model is Better to Resolve the Polysemy of Korean Postposition?

Tables 2-7 show the classification performance (i.e., *F*-score) of the two models for each postposition. Results show that the overall *F*-score was the highest in BERT (0.875; for *-eyse*) and the lowest in GPT-2 (0.676; for *-(u)lo)*).

Comparison	$\|F\|$	p
Model	752.97	$< .001^{***}$
Postposition	1240.18	$< .001^{***}$
Model $\times$ Postposition	97.14	$< .001^{***}$

Table 8: Results of the two-way ANOVA for the overall comparison of two models

Note. $^{***} < .001$

To construct a global model, we performed a two-way ANOVA (2 models $\times$ 3 postpositions). As Table 8 shows, there are significant differences in the *F*-score across the models and postpositions. This indicates the classification performance differed between the BERT model and the GPT-2 model in all postpositions.

Comparison	$\|t\|$	p
BERT vs. GPT-2 (*-ey*)	14.506	$< .001^{***}$
BERT vs. GPT-2 (*-eyse*)	9.688	$< .001^{***}$
BERT vs. GPT-2 (*-(u)lo*)	21.337	$< .001^{***}$

Table 9: Statistical comparison between two models by each postposition: Two-sample *t*-test

Note. $^{***} < .001$

We further conducted post-hoc analyses through a two-sample *t*-test. As Table 9 shows, the model performance of BERT significantly differs from the GPT-2' performance. Considering the differences between two models for model training such as the different directions or pre-training tasks of two models (see Section 2.3), this can indicate that the different training processes of the two models influenced the classification performance by classifying the functions of the postpositions.

3.2 Does the Number of Functions Involving a Postposition Affect the Model Performance?

As shown in Tables 2-7, the BERT model performed better for *-eyse*, which has only two functions (SRC and LOC), than for the other two postpositions (*-ey* and *-(u)lo*). Similar to the BERT model, *-eyse* outperformed the other two postpositions in the GPT-2 model, as Tables 2-7 show. The overall classification *F*-score rates for *-ey*, *-eyse* and *-(u)lo* were around 0.744, 0.875 and 0.795 for BERT, 0.68, 0.844 and 0.676 for GPT-2.

Comparison	$\|F\|$	p
Postposition	22.941	$< .001^{***}$
Epoch	0.414	0.521
Postposition $\times$ Epoch	0.003	0.959

Table 10: Results of the two-way ANOVA for the BERT model

Note. $^{***} < .001$

Table 10 shows the two-way ANOVA for the comparison of the BERT classification performance. The result presents that the overall *F*-score levels of the postpositions were significantly different from each other. This indicates there is a difference in model performance between the three postposition types which have a different number

of functions.

Comparison	$\|F\|$	p
Postposition	0.049	0.825
Epoch	1.690	0.196
Postposition × Epoch	0.355	0.552

Table 11: Results of the two-way ANOVA for the GPT-2 model

Unlike the results from the BERT model, the statistical analysis of two-way ANOVA for GPT-2 (Table 11) shows that there was no statistical significance in the performance across the postpositions/epochs. This indicates that GPT-2 is not affected by the number of functions of each postposition.

3.3 Do the Asymmetric Proportions of the Functions in Each Postposition Affect the Model Performance?

The answer is *they do*. For the BERT model, the overall classification *F*-score of each function for *-ey* was the highest for CRT (0.837) and the lowest for AGT (0.217); for *-eyse*, the performance was the highest in LOC (0.93) and the lowest in SRC (0.373); for *-(u)lo*, the classification performance was the highest in DIR (0.911) and the lowest in LOC (0.233) (Tables 2-4). Similar to the BERT model, the overall classification performance of GPT-2 for *-ey* was the highest in CRT (0.796) and the lowest for AGT (0.003); for *-eyse*, the *F*-score was the highest in LOC (0.912) and the lowest in SRC (0.272); for *-(u)lo*, the classification performance was the highest in DIR (0.782) and the lowest in LOC (0.226) (Tables 5-7).

As for the occurrences of individual functions per postposition, CRT for *-ey*, LOC for *-eyse*, and DIR for *-(u)lo* account for the larger portion of the entire corpus than other functions (see Table 1). This finding thus indicates that the model performance was affected by the asymmetric proportions of the functions comprising the use of each postposition.

3.4 How do the Transformer-Architecture Models Classify Sentences for Each Postposition Based on Function as the Epoch Proceeds?

Our visualization system showed that the model was able to recognize the functions of each postposition as the epoch progressed. Through the outcomes of the BERT model, for *-ey*, the number of clusters was one when the epoch was one, but as the epoch progressed, the sentences were divided into four in Epoch 8, five in Epoch 12, and six in Epoch 40. For *-eyse*, all of the sentences were grouped into one when the epoch was one, and there were two clusters since the epoch was two. For *-(u)lo*, the number of clusters increased, starting from one (Epoch 1) to four (Epoch 4), five (Epoch 9), and six (Epoch 29).

The GPT-2 model also showed a similar tendency with the BERT model. For *-ey*, all of the sentences were grouped into one when the epoch

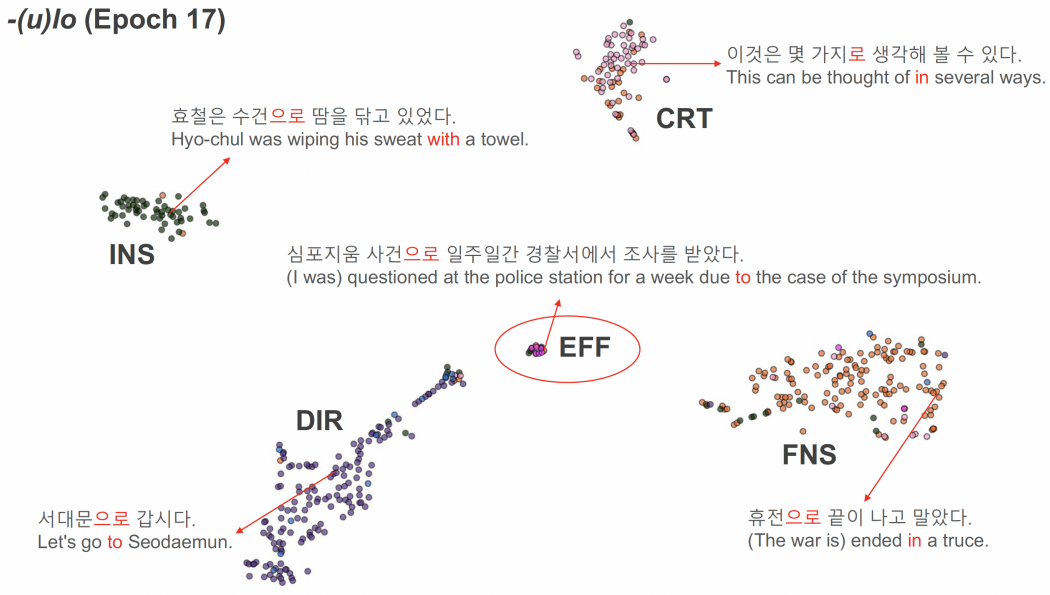

Figure 3: The t-SNE outcome of BERT model for *-(u)lo* in Epoch 17

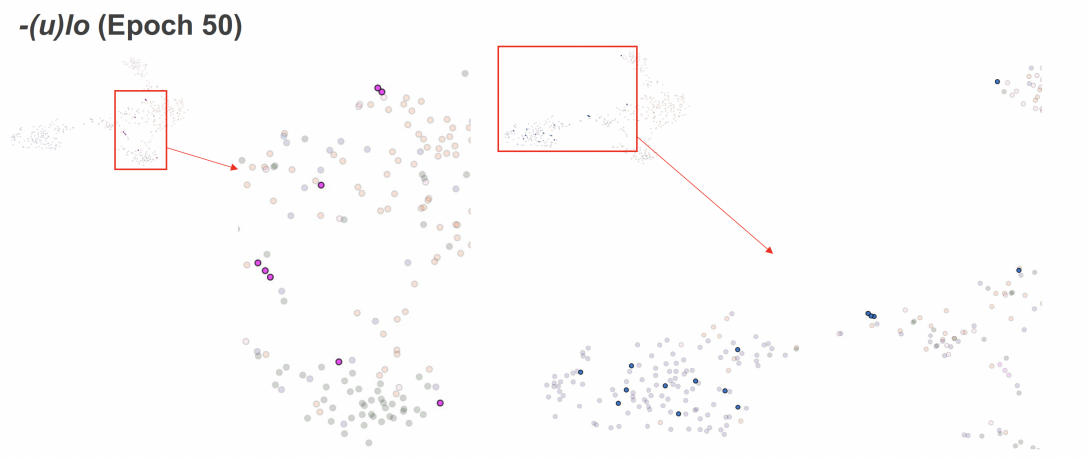

Figure 4: The t-SNE outcome of the GPT-2 model for *-(u)lo* in Epoch 50 (left: for highlighting the EFF instances; right: highlighting the LOC instances)

was one, but as the epoch progressed, the sentences were divided into two in Epoch 8, three in Epoch 23, four in Epoch 42, and five in Epoch 47. For *-eyse*, the number of clusters was one when the epoch was one, and there were two clusters since the epoch was 20. For *-(u)lo*, the number of clusters increased, starting from one (Epoch 1) to two (Epoch 3), three (Epoch 18), and four (Epoch 23).

In particular, by using visualization system, we found two interesting aspects. First, the BERT model in Epoch 17 (Figure 3) for *-(u)lo*, a cluster of EFF (the function with low-frequency occurrences in the data) emerged. This finding indicates that the BERT model can identify functions at a satisfactory level, even though they are relatively infrequent, as long as there are sufficient epochs provided. However, unlike the BERT model, the GPT-2 model did not recognize EFF as a designated function until Epoch 50 as shown in Figure 4 (left).

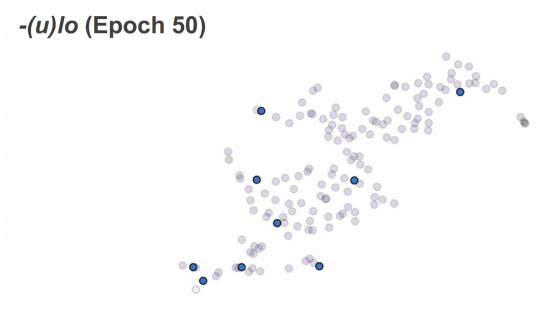

Figure 5: The DIR cluster in the t-SNE outcome of the BERT model for *-(u)lo* (Epoch 50) highlighting the LOC instances

Second, for both models, LOC could not form a designated cluster in the end. Zooming into the individual instances of LOC (Figure 4 (right) and Figure 5), we found that many of the LOC instances (11 out of 15) belonged to the DIR group. This may be due to (i) the low frequency of LOC in the data and (ii) the semantic closeness between DIR and LOC—they relate to a location and are often difficult to distinguish one from another. This finding indicates that there are still some limitations in regard to the identification of functions given the above complications.

4 Conclusion

In this study, we made five major findings. First, BERT performed better than GPT-2 in revealing the polysemy of Korean postpositions. Second, there was an inverse relation between the classification performance and the number of functions of each postposition. Third, the model was affected by the corpus size of each function. Fourth, the model was able to identify the intended functions of a postposition as the epoch progressed. Fifth, these models were affected by the rarely occurring input and/or semantic closeness between the items, limiting the performance of two models in the given task to some extent.

The findings of this study should be further verified by incorporating more postposition types that have similar degrees of polysemy that three adverbial postpositions demonstrate. Future study will also benefit from considering other contextualized word-embedding models such as GPT-3 (Brown et al., 2020) or ELECTRA (Clark et al., 2020) to better ascertain the advantage of transformer-architecture models in this kind of task.

We believe our visualization system will con-

tribute to extending the current understanding of how transformer-architecture models work for language tasks (particularly in non-English settings).

Acknowledgments

We would like to thank Gyu-Ho Shin and the three anonymous reviewers for their comments and feedback. The research of Seongmin Mun was also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1I1A2051993) and Institute for Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) (2019-0-01371, Development of brain-inspired AI with human-like intelligence).

References

Jangseong Bae, Changki Lee, Junho Lim, and Hyunki Kim. 2020a. Bert-based data augmentation techniques for korean semantic role labeling. pages 335–337.

Jangseong Bae, Changki Lee, and Soojong Lim. 2015. Korean semantic role labeling using deep learning. *Korean Information Science Society*, 6:690–692.

Jangseong Bae, Changki Lee, Soojong Lim, and Hyunki Kim. 2020b. Korean semantic role labeling with bert. *The Korean Institute of Information Scientists and Engineers*, 47(11):1021–1026.

Jangseong Bae, Junho Oh, Hyunsun Hwang, and Changki Lee. 2014. Extending korean propbank for korean semantic role labeling and applying domain adaptation technique. *Korean Information Processing Society*, pages 44–47.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Jeong-mi Cho and Gil-cheng Kim. 1996. A study on the resolving of the ambiguity while interpretation of meaning in korean. *The Korean Institute of Information Scientists and Engineers*, 14(7):71–83.

Miho Choo and Hye-young Kwak. 2008. *Using Korean*. Cambridge University Press, New York, NY.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

Guillaume Desagulier. 2019. Can word vectors help corpus linguists? *Studia Neophilologica*, 91(2):219–240.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Philip Gage. 1994. A new algorithm for data compression. *C Users J.*, 12(2):23–38.

Dylan Glynn and Justyna Robinson. 2014. *Corpus Methods for Semantics*. Corpus Methods for Semantics. Quantitative studies in polysemy and synonymy. John Benjamins.

Janosch Haber and Massimo Poesio. 2021. Patterns of polysemy and homonymy in contextualised language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2663–2676, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Seung-Yean Hong, Seung-Hoon Na, Jong-Hoon Shin, and Young-kil Kim. 2020. Roberta and stack pointer network for korean semantic role labeling. *The Korean Institute of Information Scientists and Engineers*, pages 362–364.

Heewon Jeon, Hyung jun Kim, Seujung Jung, Muhyun Kim, Yunho Maeng, Kyeongpil Kang, and Sangwhan Moon. 2021. Kogpt2 ver 2.0.

Heewon Jeon, Donggeon Lee, and Jangwon Park. 2019. Korean bert pre-trained cased (kobert).

Byong-cheol Jeong. 2010. An integrated study on the particle '*-ey*' based on the simulation model. *The Linguistic Science Society*, 55:275–304.

Wan-su Kim and Cheol-young Ock. 2016. Korean semantic role labeling using case frame dictionary and subcategorization. *The Korean Institute of Information Scientists and Engineers*, 43(12):1376–1384.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages

4801–4811, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Changki Lee, Soojong Lim, and Hyunki Kim. 2015. Korean semantic role labeling using structured svm. *The Korean Institute of Information Scientists and Engineers*, 42(2):220–226.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through WordNet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691, Florence, Italy. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Chris McCormick. 2019. Bert fine-tuning tutorial with pytorch.

George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alexander Wong. 2021. Umls-BERT: Clinical domain knowledge augmentation of contextual embeddings using the Unified Medical Language System Metathesaurus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1744–1753, Online. Association for Computational Linguistics.

Seongmin Mun. 2021. *Polysemy resolution with word embedding models and data visualization : the case of adverbial postpositions -ey, -eyse, and -(u)lo in Korean*. Theses, Université de Nanterre - Paris X.

Seongmin Mun and Gyu-Ho Shin. 2020. Context window and polysemy interpretation: A case of korean adverbial postposition *-(u)lo*. In *IMPRS Conference 2020: Interdisciplinary Approaches to the Language Sciences, Max Planck Institute for Psycholinguistics*.

Ki-sim Nam. 1993. The use of the korean postposition: focus on '*-ey*' and '*-(u)lo*'. *sekwang hakswul calyosa*.

Chanmin Park, Yeongjoon Park, Youngjoong Ko, and Jungyun Seo. 2019. Semantic role labeling using the korean elmo embedding. *The Korean Institute of Information Scientists and Engineers*, pages 608–610.

Jeong-woon Park. 1999. A polysemy network of the korean instrumental case. *Korean Journal of Linguistics*, 24(3):405–425.

Giovanni Puccetti, Alessio Miaschi, and Felice Dell'Orletta. 2021. How do BERT embeddings organize linguistic knowledge? In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 48–57, Online. Association for Computational Linguistics.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018b. Language models are unsupervised multitask learners.

Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. 1998. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL (1)*. The Association for Computer Linguistics.

Hyo-pil Shin. 2008. The 21st sejong project : with a focus on selk (sejong electronic lexicon of korean) and the knc (korean national corpus). In *In The 3rd International Joint Conference on Natural Langauge Processing*.

Myung-chul Shin, Yong-hun Lee, Mi-young Kim, Youjin Chung, and Jong-hyeok Lee. 2005. Semantic role assignment for korean adverbial case using sejong electronic dictionary. *Korea Information Science Society*, pages 120–126.

Ho-Min Sohn. 1999. *The korean language*. Cambridge University Press, Cambridge, UK.

Aina Garí Soler and Marianna Apidianaki. 2021. Let's play mono-poly: BERT can reveal words' polysemy level and partitionability into senses. *CoRR*, abs/2104.14694.

Dae-heon Song. 2014. A study on the adverbial case particles of '*-ey*' and '*-eyse*' for korean language education. *The Association of Korean Education*, 101:457–484.

Raúl Vázquez, Alessandro Raganato, Mathias Creutz, and Jörg Tiedemann. 2020. A systematic study of inner-attention-based sentence representations in multilingual neural machine translation. *Computational Linguistics*, 46(2):387–424.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Yu Wu, Wei Wu, Chen Xing, Can Xu, Zhoujun Li, and Ming Zhou. 2019. A sequential matching framework for multi-turn response selection in retrieval-based chatbots. *Computational Linguistics*, 45(1):163–197.

David Yenicelik, Florian Schmidt, and Yannic Kilcher. 2020. How does BERT capture semantics? a closer look at polysemous words. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 156–162, Online. Association for Computational Linguistics.

Zeyu Yun, Yubei Chen, Bruno Olshausen, and Yann LeCun. 2021. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 1–10, Online. Association for Computational Linguistics.