Received: 25 August 2021 | Revised: 29 July 2022 | Accepted: 29 August 2022

DOI: 10.4218/etrij.2021-0293

ORIGINAL ARTICLE

ETRI Journal WILEY

Multi-task learning with contextual hierarchical attention for Korean coreference resolution

Cheoneum Park

AIRS Company, Hyundai Motor Group, Seoul, Republic of Korea

Correspondence
Cheoneum Park, AIRS Company, Hyundai Motor Group, Seoul, Republic of Korea.
Email: parkce3@gmail.com

Abstract

Coreference resolution is a task in discourse analysis that links several headwords used in any document object. We suggest pointer networks-based coreference resolution for Korean using multi-task learning (MTL) with an attention mechanism for a hierarchical structure. As Korean is a head-final language, the head can easily be found. Our model learns the distribution by referring to the same entity position and utilizes a pointer network to conduct coreference resolution depending on the input headword. As the input is a document, the input sequence is very long. Thus, the core idea is to learn the word- and sentence-level distributions in parallel with MTL, while using a shared representation to address the long sequence problem. The suggested technique is used to generate word representations for Korean based on contextual information using pre-trained language models for Korean. In the same experimental conditions, our model performed roughly 1.8% better on CoNLL F1 than previous research without hierarchical structure.

KEYWORDS

coreference resolution, hierarchical model, head-final language, multi-task learning, pointer network

1 | INTRODUCTION

Coreference resolution [1,2] is a natural language processing (NLP) task that identifies and links words used in different entity representations. This is useful when an entity is represented by alternative words, aliases, acronyms, pronouns, determiners, and other substitutes. The information about the entity mentioned in the discourse or document can be maintained consistently and accurately if the reference relationship between the words used as other expressions can be correctly determined. Therefore, coreference resolution is critical in understanding the entities that appear in documents. It is a prerequisite for numerous higher-level NLP tasks used in natural language understanding, including question answering (QA), document summarization, machine translation, and information extraction.

The span ranking-based model in English identifies and ranks candidate spans in documents to perform coreference resolution, such as e2e-coref [3] and c2f-coref [4], and the model has achieved significant performance. However, because it extracts all spans from a document and performs ranking for all spans with an attention mechanism [5] the span ranking model has a time complexity of $O(n^4)$. The head of a phrase determines the syntactic structure and is a word that holds the meaning of the phrase. As Korean is a head-final language, the head always appears at the end of every phrase. Mention

detection, which is a candidate for coreference resolution, is performed using phrases, and the head-final characteristics are reflected as is.

We define only nouns from a document as candidates based on the head-final feature and perform ranking using a pointer network (Ptr-net) [6] for the input document. The time complexity is $O(n^2)$ since our model only needs to consider the number of words and nouns in the input document. Ptr-net, an extended model of the recurrent neural network (RNN) encoder-decoder [7], uses an attention mechanism to output positions corresponding to an input sequence. Ptr-net can extract information by pointing out a specific word in an input document or a sentence.

Multi-task learning (MTL) [8] is a method of learning several tasks simultaneously using a single model. Because a model shares the information about related tasks, MTL can effectively generalize the tasks, which subsequently have a positive effect on one another. When the data are limited, it is difficult for the model to determine which features are relevant and which are not. Furthermore, MTL provides additional evidence to distinguish the relevance of features by focusing attention on important features in other tasks [9]. Our model classifies words that refer to the same entity based on the head. We train the task of classifying the start boundary of the mention and the mention information is shared with the model parameters. Furthermore, we train the sentence classification task alongside the coreference resolution as an auxiliary task.

If the input sequence consists of several sentences, the length of the input sequence becomes excessive, and the performance is degraded. The hierarchical structure model is more robust for long-term sequences because it generates a context vector by encoding an input document using both word- and sentence-level modeling. We perform sentence-level encoding and calculate contextual hierarchical attention at the word and sentence levels to solve the problem of decreasing performance as the document input sequence becomes longer. Each task is defined as a separate task, and MTL is used to reflect the context information of the word and sentence levels calculated in the coreference resolution to shared parameters.

Our contributions are summarized as follows:

- We use hierarchical architecture to encode sentence-level context on Korean coreference resolution.
- We use MTL loss functions for word- and sentence-levels to melt coreferent sentence information to word level.
- We achieve a higher CoNLL F1 score than a previous study without hierarchical structure.

2 | RELATED WORKS

Existing coreference resolution studies can be classified as rule-, statistical-, and combined rule-/statistical-based methods. Stanford's model [10], which was a rule-based method applied to multi-level rules using pronouns, entity attributes, and named entity information [11], was used to define a multi-sieve suitable for Korean coreference resolution. The mention-pair model [12] was a statistically based method for pairing present-mention and arbitrary antecedents, which used machine learning to solve coreferences. The method presented in Park and others [11] was a model that used feed-forward neural networks to combine rule- and statistical-based coreference resolutions. In Lee and others [13], end-to-end pronominal coreference resolution was performed using Ptr-net. However, the performance suffered on a document with long sentences. Furthermore, Ptr-net was only used on pronouns. Clark and Manning [14] trained an entity-clustering model using the RNN and solved the coreference problem. In Clark and Manning [15], coreference resolution was performed using reinforced learning to rank mentions.

2.1 | Head-final coreference resolution

Language has the head-directional characteristic and is composed of two structures: (1) right-branching head-initial structures and (2) left-branching head-final structures. English has a head-initial structure, whereas Korean and Japanese have a head-final structure. In Chinese, noun phrases have a head-final structure, but some verb phrases have a head-initial feature. Models such as e2e-coref and c2f-coref are based on span ranking to perform coreference resolution in English, which is a head-initial language. Currently, the span ranking model assumes that all spans can be extracted from a document containing n words as candidates; thus, the span ranking model has $S = (n(n+1))/2 = O(n^2)$ spans. Hence, the search space of the span ranking-based coreference resolution model is $(S(S+1))/2 = O(n^4)$ because the model performs head attention to identify the head in the span and ranks the extracted candidate spans. However, in Korean, the head is always placed at the end of a noun phrase because it is a head-final language. We assume that all nouns are candidates for coreference resolution using the head-final feature, and identify and link words that represent the same entity as the candidate in the input document. We use a Ptr-net based on the attention mechanism, and the coreference resolution search space of our model is $O(nm)$, where n is the number of words and m is the number of nouns.

2.2 | Hierarchical encoder-decoder

A hierarchical encoder-decoder (HRED) model was used to perform model training and prediction by generating not only word-level input sequences but also sentence-level encodings, which are the upper layers of the input sequence [16]. The HRED model included a word-level encoder RNN, a sentence-level encoder RNN, and an output-producing decoder RNN. The sentence-level encoder RNN was generated from the hidden state of the word-level encoder RNN. The decoder RNN performed training and prediction using the hidden encoding state of both the word and sentence encoders. The HRED model was used in Serban and others [17] to convert the token unit of the input sequence into word-level encoding. The following token (i.e., the encoder input sequence) was converted into sentence-level encoding. The encodings were subsequently applied to a dialog model. The method in Sordoni and others [18] used the HRED model to generate query-level (i.e., word-level) encoding from words in a query in the encoder input sequence and then applied session-level (i.e., sentence-level) encoding to the information retrieval model for each query representation. In Lin and others [19], language modeling for a document was performed using the HRED model. The HRED model was used to create a word- and sentence-level encodings in [20], which were applied to document summarization.

2.3 | Multi-task learning

As with transfer and zero-shot learning, MTL has been used extensively in machine learning to improve task generalization using data containing multiple tasks, because it can train the knowledge transfer process as a single model. MTL has been applied to web searches in NLP. Furthermore, the work of McCann and others [21] and Xu and others [22] demonstrated successful results in the machine reading comprehension task, which understands input documents and questions, and finds answers. Luong and others [23] proposed an MTL method for translating tasks based on a sequence-to-sequence model. Among these, the encoder is shared among several tasks such as machine translation and low-level parsing in the one-to-many setting, demonstrating that MTL is mutually beneficial.

3 | PROPOSED MODEL

Ptr-net is a model that outputs a position corresponding to the input sequence using the RNN encoder-decoder, based on the attention mechanism. It can solve problems with variable-length output classes.

We model word- and sentence-level encoders using a pre-training language model (PLM) with a large-capacity corpus and a decoder, as shown in Figure 1. As shown in Figure 1, our model generates a word-level representation from the PLM's contextual information regarding the document input and encodes the generated hidden state with the RNN. In this case, we extract the vector of the position corresponding to the end of the sentence to create a sentence-level hidden state and encode the extracted vectors with the RNN to create a sentence-level representation.

The decoder input is formed by the extraction of the hidden state corresponding to the noun position in the word-level representation. The extracted hidden state is encoded using the RNN. The decoder uses a gated self-attention network [26] to calculate the alignment score and reflect the relationship information from the decoder sequence. Thereafter, coreference resolution, mention start boundary classification, and sentence classification with the word-referenced entity is performed using Ptr-net in the output layer. We use a biaffine attention [27] as the attention score function.

3.1 | Model input

3.1.1 | Task formulation

We define coreference resolution in Korean as a classification task based on the ranking score of Ptr-net at the document level. When a document is defined as D, it contains N_D sentences $D = \{s_1, s_2, s_3, ..., s_{N_D}\}$. Each sentence S consists of N_S words and $s = \{x_1, x_2, x_3, ..., x_{N_S}\}$. We generate a context vector based on the PLM in Figure 1.

The training dataset consists of the word-level encoder input sequence $X = \{x_1, x_2, ..., x_n\}$, the encoder input feature sequence $F = \{f_1, f_2, ..., f_n\}$, the decoder input noun sequence $U = \{u_1, u_2, ..., u_m\}$, the sentence-level encoder input sequence $P = \{p_1, p_2, ..., p_l\}$, the coreference resolution output sequence $Y_a = \{a_1, a_2, ..., a_m\}$, the mention detection output sequence $Y_b = \{b_1, b_2, ..., b_m\}$, and the sentence classification output sequence $Y_e = \{e_1, e_2, ..., e_l\}$. In the above, n is the length of the word-level encoder input sequence, m is the length of the candidate sequence for the coreference resolution extracted from the input document, and l is the length of the sentence-level encoder input sequence.

Example of input sequence of encoder and decoder

In the following example, we present the input sequence as the encoder input and extract nouns that are

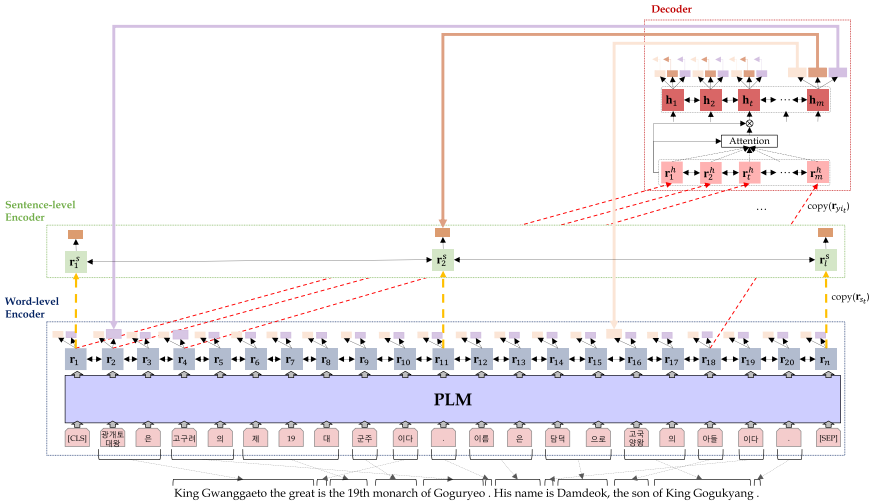

FIGURE 1 RNN encoder-decoder-based Ptr-net with MTL for contextual hierarchical attention. The input is the application of byte pair encoding (BPE) to the result of the morpheme analysis. The encoder consists of word- and sentence-level encodings, and the PLM is BERT [24] or ELECTRA [25]. The sentence-level input is a copy of the vector corresponding to the end of the sentence from the hidden state created by the word-level encoder. The yellow dashed arrow line indicates the copy of the sentence vector. The decoder input is a copy of the vector corresponding to the noun position from the encoder. The red dashed arrow line represents the decoder input, which is a copy of the noun vector. The decoder output is the position of a word with the same entity meaning as the input word, the start boundary of the mention, and the position of the sentence containing the referenced word

coreference resolution candidates to create a noun sequence. The noun sequence is fed into the decoder as input.

In Figure 1, the example of the input sequence and noun sequence is as follows:

- Input sequence: 19. .(King Gwanggaeto the great is the 19th monarch of Goguryeo. His name is Damdeok, the son of King Gogukyang.)
- Noun sequence: , , 19, , , , , , (King, Gwanggaeto, great, 19th, monarch, Goguryeo, His, name, Damdeok, son, King, Gogukyang)

3.1.2 | BERT style input format

We assume that all nouns are candidates for coreference resolution because we design a head-final structure-based model and perform coreference resolution for the head. The BERT approach used is a pre-trained model from the electronics and telecommunications research institute (ETRI),[1] and the BERT vocabulary is BPE applied to morphological analysis results.

An example follows:

- Raw sentence: 1910 3 .(Gyeongcheon is an article left by martyr An Jung-geun in March 1910 ahead of his execution.)
- Morphological analyzed sentence with POS-tag: /NNP /JX 1910/SN /NNB 3/SN /NNP /NNP /NNG /JKS /NNG /NNG /JKO /VV /EC /VV /ETM /NNG /VCP /EF ./SF
- Applied BPE: /NNP_ /JX_ 19 10/SN_ /NNB_ 3/SN_ /NNP_ /NNP_ /NNG_ /JKS_ /NNG_ /NNG_ /JKO_ /VV_ /EC_ /VV_ /ETM_ /NNG_ /VCP_ /EF_ ./SF_

As in the previous example, we perform morphological analysis on the raw sentence and add the morpheme and part-of-speech (POS) tag to make it resemble "/NNP" (An Jung-geun). We do not associate the POS tag with the morpheme when using ELECTRA. Thereafter, we use BPE with the sequence separated by subwords as the model input. We apply BPE to one morpheme at a time and identify the morphemes by appending _ to the final token of the divided subwords.

Accordingly, we perform morphological analysis on the input document before using it as model input with BPE. The BERT pre-trained data input format [CLS] is added as the first token of the input sequence and [SEP] as the final token. The output results in Figure 1 are the position of the antecedent in the coreference resolution, the starting boundary of the mention, and the position of the sentence containing the antecedent. A location to be resolved is pointed to and output as a result if it exists as a coreferent, mention boundary, or sentence position. Otherwise, the position of [CLS], the first token in the input sequence, is output. The first token among the tokens applying BPE to the morpheme

[1](http://aiopen.etri.re.kr/)

is defined as the pointing criterion to point to the position of the coreference resolution entity or the mention start boundary.

3.1.3 | Input features

We use the following features for the coreference resolution: word boundary (*word*), morpheme boundary (*morp*), dependency parsing (*dep*), named-entity recognition (*NER*), and document type (*dtype*).

The features are described as follows:

- Word boundary (*word*): We use the word boundary feature to learn the word range characteristics of the coreference resolution expressed in word units. The word starting token is denoted by the B tag and the subsequent token is denoted by the I tag.
- Morpheme boundary (*morp*): We use the morpheme boundary feature to reflect the morpheme range characteristics of the morphological analysis result. The starting token of the morpheme is morp-B, and the following tokens are classified with the same tag as morp-I. When using a PLM like ELECTRA, we use the POS tag attached to the morpheme boundary tag.
- Dependency parsing (*dep*): We use the dependency parsing label information as a feature to reflect the structural and semantic information of the input sentences. The B and I tags are attached in the same way as for the word boundary feature because dependency parsing is a word unit.
- Named-entity recognition (*NER*): We use the type information for each named entity that appears as a feature in the document. The BIO tag is used by the NER.
- Document type (*dtype*): This feature determines the type of a given document, which we define as question and answer.

3.2 | Model architecture

3.2.1 | Encoder

Word level

Following the morphological analysis, the model is fed an input sequence tokenized with BPE. Our PLM-based model generates token, segment, and position embeddings for the input sequence and combines them. Thereafter, the model uses the transformer to generate hidden states for the PLM input sequence, concatenates the hidden state of the PLM output with the input feature representation, and encodes it as a bidirectional simple recurrent unit (biSRU) [28]. The biSRU generates $\mathbf{r}_i$ as in (1), where $\mathbf{e}_i$ i the hidden states concatenated to the PLM and the feature representation's output-hidden states.

$$\mathbf{r}_i = \text{biSRU}(\mathbf{r}_{i-1}, \mathbf{e}_i). \tag{1}$$

Sentence level

As indicated in Figure 1, we extract the vector to be used in the sentence-level encoder from the hidden states generated by the word-level encoder. Assuming that the vector of each sentence's final position is a sentence representation, we generate the sentence-level hidden states $\mathbf{r}_{p_j}$ by sentence encoding with biSRU(.). The sentence-level hidden states $\mathbf{r}_j^s$ are generated using (2).

$$\mathbf{r}_j^s = \text{biSRU}(\mathbf{r}_{j-1}^s, \mathbf{r}_{p_j}). \tag{2}$$

3.2.2 | Decoder

In Figure 1, the decoder input is $\mathbf{r}_{u_t}$, which extracts the hidden state corresponding to the noun u_t from the word-level encoded hidden states $\mathbf{r}_i$. We model the contextual information between nouns using biSRU(.) in the decoder, as indicated in (3).

$$\mathbf{r}_t^h = \text{biSRU}(\mathbf{r}_{t-1}^h, \mathbf{r}_{u_t}). \tag{3}$$

Self-Attention Module

We apply a gated self-matching layer to model the scores between similar heads, with the following equations:

$$\mathbf{h}_t = \text{biSRU}(\mathbf{h}_{t-1}, \mathbf{g}_t), \tag{4}$$

$$\mathbf{g}_t = \text{sigmoid}(\mathbf{W}_g[\mathbf{h}_t^h; \mathbf{c}_t]) \odot [\mathbf{h}_t^h; \mathbf{c}_t], \tag{5}$$

$$\mathbf{c}_t = \sum_{k=1}^{m} \alpha_{t,k} \mathbf{h}_t^h, \tag{6}$$

$$\alpha_{t,k} = \exp(\mathbf{h}_k^h \mathbf{W}_\alpha \mathbf{h}_{t'}^h) / \sum_k \exp(\mathbf{h}_k^h \mathbf{W}_\alpha \mathbf{h}_t^h), \tag{7}$$

where $\mathbf{c}_t$ is the context vector of all nouns and $\mathbf{g}_t$ is a vector generated from an additional gate. The additional gate

concatenates the context vector $\mathbf{c}_t$ and the hidden state $\mathbf{h}_t^h$, then uses a sigmoid gate to convert the significant values of the two vectors into larger values and the insignificant values into smaller ones. The decoder models $\mathbf{g}_t$ with the gate applied and $\mathbf{h}_t$ generated by biSRU(.).

Deep Biaffine score

We apply elu [29] to the final hidden states $\mathbf{h}_t$ of the decoder, as in Dozat and Manning [27], to output the coreference resolution, mention start boundary, and sentence classification, and the decoder generates the hidden states as $\mathbf{h}_t^{\text{coref\_src}}$, $\mathbf{h}_t^{\text{men\_src}}$, and $\mathbf{h}_t^{\text{sent\_src}}$. The hidden states to be used for the coreference resolution and mention boundary outputs in this case are $\mathbf{h}_i^{\text{coref\_tgt}}$ and $\mathbf{h}_i^{\text{men\_tgt}}$, respectively, based on the output hidden state $\mathbf{r}_i$ of the word-level encoder. To determine the sentence classification, our model compares the hidden state $\mathbf{h}_i^{\text{sent\_tgt}}$ to the hidden states $\mathbf{r}_j^s$ of the sentence-level encoder.

$$\mathbf{h}_t^{\text{coref\_src}} = \text{elu}(\text{FFNN}^{(\text{coref\_src})}(\mathbf{h}_t)), \tag{8}$$

$$\mathbf{h}_i^{\text{coref\_tgt}} = \text{elu}(\text{FFNN}^{(\text{coref\_tgt})}(\mathbf{r}_i)), \tag{9}$$

$$\mathbf{h}_t^{\text{men\_src}} = \text{elu}(\text{FFNN}^{(\text{men\_src})}(\mathbf{h}_t)), \tag{10}$$

$$\mathbf{h}_i^{\text{men\_tgt}} = \text{elu}(\text{FFNN}^{(\text{men\_tgt})}(\mathbf{r}_i)), \tag{11}$$

$$\mathbf{h}_t^{\text{sent\_src}} = \text{elu}(\text{FFNN}^{(\text{sent\_src})}(\mathbf{h}_t)), \tag{12}$$

$$\mathbf{h}_j^{\text{sent\_tgt}} = \text{elu}(\text{FFNN}^{(\text{sent\_tgt})}(\mathbf{r}_j^s)). \tag{13}$$

We use the deep biaffine score to output the coreference resolution, mention boundary, and sentence classification when using the attention with the following equations:

$$s_{t,i}^{\text{coref}} = \mathbf{h}_i^{\top\,\text{coref\_tgt}} \mathbf{U} \mathbf{h}_t^{\text{coref\_src}} + \mathbf{w}^\top \mathbf{h}_t^{\text{coref\_src}}, \tag{14}$$

$$s_{t,i}^{\text{men}} = \mathbf{h}_i^{\top\,\text{men\_tgt}} \mathbf{U} \mathbf{h}_t^{\text{men\_src}} + \mathbf{w}^\top \mathbf{h}_t^{\text{men\_src}}, \tag{15}$$

$$s_{t,j}^{\text{sent}} = \mathbf{h}_j^{\top\,\text{sent\_tgt}} \mathbf{U} \mathbf{h}_t^{\text{sent\_src}} + \mathbf{w}^\top \mathbf{h}_t^{\text{sent\_src}}. \tag{16}$$

Loss Function for MTL

We use MTL, which learns both word- and sentence-level classification, to use sentence-level contextual information when performing coreference resolution. The outputs of our model are coreference resolution, mention detection, and sentence classification, for which three losses $\alpha\mathcal{L}_{\text{coref}}$, $\beta\mathcal{L}_{\text{men}}$, and $\gamma\mathcal{L}_{\text{sent}}$, respectively, are calculated and summed.

TABLE 1 Dataset statistics of ETRI quiz domain for Korean coreference resolution

	Training	Development	Test
Document	2819	645	571
Sentence	8299	1086	1167
Word	126720	12834	14334
Mention	30923	1978	2431
Entity	10416	799	931

$$\mathcal{L}_{\text{coref}} = -\sum_{i}^{n} y_i^{\text{coref}} \log \hat{y}_i^{\text{coref}}, \tag{17}$$

$$\mathcal{L}_{\text{men}} = -\sum_{i}^{n} y_i^{\text{men}} \log \hat{y}_i^{\text{men}}, \tag{18}$$

$$\mathcal{L}_{\text{sent}} = -\sum_{j}^{l} y_j^{\text{sent}} \log \hat{y}_j^{\text{sent}}. \tag{19}$$

Equation (20) calculates the final loss for the MTL by adding the cross-entropy loss [30]. Here, α,β, and γ are hyperparameters for performing optimization.

$$\mathcal{L} = \alpha\mathcal{L}_{\text{coref}} + \beta\mathcal{L}_{\text{men}} + \gamma\mathcal{L}_{\text{sent}}. \tag{20}$$

4 | EXPERIMENTS

4.1 | Datasets and measurements

In this study, we use a coreference resolution dataset [31] from the ETRI quiz domain, which consists of Janghak-quiz and wiseQA. The document statistics for the ETRI datasets is presented in Table 1. We use the CoNLL F1 averaged MUC, B^3, and CEAF_{ϕ_4} according to the official CoNLL-2012 evaluation script.[2] However, because the script is suitable for English, we evaluate the coreference resolution using only the head of the mention.

[2] https://conll.cemantix.org/2012/software.html
[3] https://github.com/google-research/bert

TABLE 2 Experimental results on a test set of Korean data from ETRI wiseQA

Model	MUC			B^3			$CEAF_{\phi_4}$			CoNLL
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
rule-based [11]	52.0	45.1	48.3	51.6	45.4	48.3	53.8	51.9	52.9	49.8
e2e-coref [3]	66.9	55.2	60.5	64.5	53.1	58.2	66.1	53.9	59.4	59.4
c2f-coref [4]	68.3	56.4	61.8	59.0	53.4	59.0	66.4	54.4	59.8	60.2
bert-coref [32]	71.7	65.0	68.2	69.3	63.0	66.0	72.2	62.4	66.9	67.0
fast-coref [33]	67.3	67.3	67.3	64.8	65.3	65.1	69.5	63.5	66.3	66.2
fast-coref w/ KD [33]	68.0	68.2	68.1	65.6	66.0	65.8	71.1	62.9	66.7	66.9
simple-coref [34]	66.4	65.5	65.9	63.3	64.6	64.0	63.8	64.6	64.2	64.7
simple-coref w/ aug. [34]	70.7	67.4	69.0	68.5	64.8	66.6	72.1	63.3	67.4	67.7
Our model	68.4	66.8	67.6	66.1	64.4	65.3	70.3	63.1	66.5	66.5

Note: The final column (CoNLL Avg. F1) is the main evaluation metric, which is calculated by averaging the F1 of MUC, B^3, and $CEAF_{\phi_4}$. We calculate the coreference resolution score based on the head of the mentions because Korean is the head-final language.

4.2 | PLM for Korean

We use BERT and ELECTRA as encoders to construct coreference resolution models with Ptr-net.

BERT consists of a bidirectional Transformer encoder with several layers. We use KorBERT, distributed by ETRI, as the baseline model and multilingual BERT[3] released by Google for the comparison experiment. Multilingual BERT uses a model[4] that ports TensorFlow to PyTorch from Hugging Face. The hyperparameters of KorBERT are the same as those of the base model of Devlin and others [24]. KorBERT is trained by Wikipedia and news data from the web, totaling 23.5 G. KorBERT generates a BPE word vocabulary by combining morphemes and POS tags and then performs morphological analysis on the training data. The dictionary consists of 30 349 BPE tokens.

We use an ETRI language analyzer, which is available from AIOpen, for the morpheme analysis. The language analyzer is a tool for Korean NLP.

Moreover, we use the Korean ELECTRA model, namely KoELECTRA.[5] KoELECTRA is pre-learned with 14 G of Korean text (96 M sentences, 2.6 B tokens). KoELECTRA uses WordPiece the vocabulary created in the EnlipleAI. PLM.[6]

4.3 | Implementations

Hyperparameter tuning is performed on the proposed model. The optimized hyperparameters of the model are as follows: The RNN type uses SRU, with the dropout set to 0.1, the number of hidden layer stacks set to 2, and the number of hidden layer dimensions and feature embedding dimensions set to 400 and 100, respectively. The number of dimensions of the biaffine score function is set to 50. The learning rate is set to 5×10^{-5} because the PLM is fine-tuned. The learning algorithm uses Adam [35], with a weight decay of 1×10^{-2}. We fine-tune all models on the ETRI Korean data for 70 epochs with a batch size of 10 for each GPU, and we limit the maximum length of the input sequence to 430 when using KorBERT and 440 when using KoELECTRA. The ETRI language analyzer is used to obtain results such as POS tagging, NER, and dependency parsing, which we then use as features.

5 | RESULTS

5.1 | Overall performance

In Table 2, we compare the proposed model to previous systems for Korean coreference resolution, such as the rule-based model [11], end-to-end neural model (e2e-coref) [3], and higher-order model (c2f-coref) [4]. Furthermore, we compare previous studies that used the same dataset, such as bert-coref [32], fast-coref [33], and simple-coref [34].

The fast-coref and simple-coref in the fifth and seventh rows of Table 2 are the results of testing with a single model in the same environment. The fast-coref w/ KD in the sixth row is the result of performing knowledge distillation (KD) using the ensemble proposed in Park and others [33], and the simple-coref w/ aug [34] in the eighth row is the result of training after dividing the document into two sentences and performing augmentation (aug).

[4]https://github.com/huggingface/transformers
[5]https://github.com/monologg/KoELECTRA
[6]https://github.com/enlipleai/kor_pretrain_LM

As a result of the experiment, we achieve a 66.5% CoNLL F1 score that is higher than the recent studies fast-coref and simple-coref in the same environment of a single model. In particular, our model's CoNLL F1 score is 1.8 higher, although the word-level coreference pointing method of simple-coref is similar to the proposed method. Our model also outperforms the e2e- and c2f-coref models, which both use only word-level information. Because sentence-level context information can be melted into the encoder during training, it can be seen that the coreference resolution is useful when the task of finding the position of the sentence, including the antecedent for the current candidate, and the coreference resolution is performed with MTL. The proposed method, in contrast, performs similarly to Joshi and others [32], which performed well in the English model, but its time complexity [32] is $O(n^4)$. The time complexity is high compared to our model as $O(n^2)$.

6 | ANALYSIS

We optimize the model's hyperparameter as proposed in the development set of the ETRI coreference resolution. We conduct various analyses, including a comparison of the PLM and training algorithms, ratio loss optimization, and ablation based on the optimized hyperparameters.

6.1 | Hyperparameter tuning

Hyperparameter tuning is carried out for the number of dimensions of the feature embedding, the number of dimensions of the hidden layer, the number of dimensions of the deep biaffine score function, the RNN type, and the number of stacks of the RNN hidden layer. We set the number of dimensions to 50, 100, 200, 400, 800, and 1600, respectively, to identify the hyperparameters that result in the best performance. The initial hyperparameters of our model are as follows: The number of dimensions of the feature embedding is set to 1600, the number of dimensions of the SRU hidden layer is set to 800, and the number of dimensions of the biaffine score function is set to 50. We use a two-layer SRU for the initial RNN type.

6.1.1 | Optimization of feature embedding dimension size

We optimize the number of dimensions of the feature embedding in the development set, as shown in Table 3. According to the experimental results, when the number of dimensions of the feature embedding is 100, CoNLL F1 performs best at 69.85%, and when the number of dimensions is 400, F1 yields a value of 69.72%. We set the number of dimensions of the feature embedding to 100 and then proceed with the hyperparameter tuning.

6.1.2 | Optimization of SRU Hidden Layer Dimension Size

As indicated in Table 4, we also optimize the number of hidden layer dimensions of the SRU. Here, when the dimension size of the hidden layer is 400, CoNLL F1 is 70.16%, which is 0.31% better than the initial value of 100.

6.1.3 | Optimization of deep biaffine score function dimension size

The dimension size optimization for the biaffine score function is shown in Table 5. The biaffine score function performs the best performance with CoNLL F1 of 70.16%

TABLE 3 Optimizing number of dimensions of feature embedding on Korean ETRI development set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
Dim.	F1	F1	F1	Pre.	Rec.	Avg. F1
50	69.93	67.72	68.26	66.23	71.25	68.64
100	70.98	68.68	69.89	68.67	71.12	**69.85**
200	70.20	68.26	69.73	67.92	70.97	69.40
400	70.73	68.50	69.93	67.79	71.80	69.72
800	70.55	68.16	68.87	67.45	71.07	69.19
1600	70.66	68.18	69.74	68.33	70.85	69.53

Note: The number of dimensions is Dim.

TABLE 4 Optimization of SRU hidden layer dimension size on Korean ETRI development set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
Dim.	F1	F1	F1	Pre.	Rec.	Avg. F1
50	70.13	68.32	70.29	66.67	72.83	69.58
100	70.98	68.68	69.89	68.67	71.12	69.85
200	71.10	68.63	70.02	69.10	70.83	69.92
400	71.20	69.02	70.26	67.46	73.11	**70.16**
800	70.55	68.17	69.11	67.65	71.02	69.28
1600	70.99	68.56	69.42	67.77	71.71	69.66

Note: The number of dimensions is Dim.

when the number of dimensions of the score function hidden layer is 50, and in this case, the coreference resolution tends to be more effective when the number of dimensions of the hidden layer is smaller.

6.2 | Comparison of PLMs

Our model generates a word representation based on the PLM output. BERT and ELECTRA have been released as Korean PLMs, and we use KorBERT developed by ETRI, Google's multilingual BERT, and the open-source KoELECTRA. The experimental results show that KorBERT has the best performance and that multilingual BERT has the lowest. KorBERT is pre-trained with POS tags combined with morphemes, so it is expected that Korean characteristics are well reflected, resulting in a good performance. The reason KoELECTRA performs worse than KorBERT is due to the difference in data size used for pre-training and the absence of POS tags, as is the case with multilingual BERT (Table 6).

6.3 | Ratio Optimization of Loss for MTL

We use MTL in conjunction with sentence classification to solve the coreference resolution task, which is processed in units of documents. The sentence classification task, defined as an auxiliary task in our model outputs the position of the sentence containing the antecedent as hierarchical attention to the sentence's contextual information. When performing MTL, the loss of each task is calculated and added together to produce a single value. The ratio to reflect the loss for each task is defined at this point, and we conduct experiments by defining the ratio as shown in Table 7. The ratio corresponds to the order of α,β, and γ used in (20). The experimental results indicate that the best performance is achieved when the ratio is 9:1:1.

TABLE 5 Optimization of deep biaffine score function dimension size on Korean ETRI development set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
Dim.	F1	F1	F1	Pre.	Rec.	Avg. F1
50	71.20	69.02	70.26	67.46	73.11	**70.16**
100	70.47	68.09	69.66	68.53	70.35	69.41
200	70.82	68.76	70.28	67.08	73.11	69.95
400	70.05	67.58	69.20	66.26	71.92	68.94
800	70.26	67.95	69.45	66.14	72.67	69.22
1600	69.58	67.56	69.01	69.55	67.93	68.72

Note: The number of dimensions is Dim.

TABLE 6 Comparison of PLMs on Korean ETRI test set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
PLM type	F1	F1	F1	Pre.	Rec.	Avg. F1
KorBERT	67.34	65.15	66.25	**67.04**	**65.54**	**66.25**
M.L.BERT	64.39	62.23	63.37	61.95	64.84	63.33
KoELEC.	65.78	63.87	65.38	65.06	65.04	65.01

6.4 | Ablation study

We propose a model that combines sentence-level attention and MTL to perform Korean coreference resolution. An ablation study is conducted to determine whether the proposed model is suitable for coreference resolution. Experiments for each component as well as the features are conducted for the ablation.

6.4.1 | Component ablation

Our model computes the results of word-level pointing and is trained with MTL and sentence-level attention. We not only calculate and output the sentence-level attention immediately but also add the score to the word-level attention score to explicitly reflect the sentence-level contextual information when searching for antecedents. In Table 8, the method as *join-score* adds the word- and sentence-level attention scores together to output the antecedent position. The experimental results show a 64.98% F1 score, which is lower than our model

TABLE 7 Ratio optimization of loss for MTL on Korean ETRI test set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
Ratio	F1	F1	F1	Pre.	Rec.	Avg. F1
6:1:3	66.12	64.06	65.11	65.78	64.52	65.10
6:3:1	66.15	63.88	65.1	66.73	63.53	65.04
6:2:2	66.58	64.55	65.41	**68.02**	63.28	65.51
7:1:2	66.7	64.02	64.36	67.74	62.65	65.03
7:2:1	66.51	64.36	65.56	65.49	65.54	65.48
8:1:1	64.93	63.05	64.36	65.10	63.19	64.11
9:1:1	67.34	65.15	66.25	67.04	**65.54**	**66.25**
10:1:1	66.39	64.54	66.10	66.84	64.58	65.68

performance. When the sentence-level module, as *s-module*, is removed from our model, a difference of 1.47% from the performance of 64.78% for F1 is observed. Accordingly, it is considered that the method of calculating the sentence-level attention in the decoder and training the output result with MTL is effective.

Feature Ablation

We use five features: morpheme boundary, word boundary, dependency parsing, NER, and head distance. We use feature ablation to observe the extent to which each feature affects our model. The NER feature in Table 9 has a performance decrease of less than 1% to 65.29% for F1, and the dependency parsing and morpheme boundary exhibit slight performance differences of 1.14% and 1.19%, respectively. The head distance feature has a value of 64.61% for F1, indicating a degradation of 1.64%, demonstrating that the distance feature is meaningful when modeling coreference resolution. Finally, the word boundary feature had a significant performance difference of 3.21%. This is because the word boundary feature replaces the meaning information of the word that is lost when the input word is tokenized and divided into subwords.

6.5 | Quality analysis

Our model is hierarchical and was built using coreference resolution and MTL training. The candidate input

TABLE 8 Component ablation study on Korean ETRI test set (best performance in bold)

	MUC	B^3	$CEAF_{\phi_4}$	CoNLL		
Model	**F1**	**F1**	**F1**	**Pre.**	**Rec.**	**Avg. F1**
Our model	67.34	65.15	66.25	**67.04**	**65.54**	**66.25**
+ join-score	65.65	63.93	65.36	64.62	65.39	64.98
− s-module	66.06	63.77	64.51	68.14	61.85	64.78

TABLE 9 Feature ablation study on Korean ETRI test set (best performance in bold)

Feature	CoNLL Avg. F1	Δ
Our model	**66.25**	
− NER	65.29	−0.96
− dependency parsing	65.11	−1.14
− morpheme boundary	65.06	−1.19
− head distance	64.61	−1.64
− word boundary	63.04	−3.21

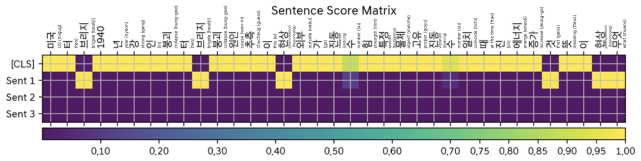

FIGURE 2 Sentence score visualization. The tokens of the candidates are on the *x* axis, and the sentence indices are on the *y* axis. Each token has an index written in front of it, with the English and pronunciation written beneath it

to the model computes the sentence level score and outputs the result. The sentence classification score for each of the given candidates is shown in Figure 2.

The entities, which are ground truth, solved coreference in the document are **`entity 1`**: { *(Tacoma), (Tacoma)*}, **`entity 2`**: { *(cause), (phenomena), (that), (phenomenon), (what)*}, **`entity 3`**: { *(bridge), (bridge)*}. At this time, the model correctly outputs all of **`entity 2, 3`** except for **`entity 1`**.

According to Figure 2, we can see that the words in the coreference relationship are highly scored among the given candidates because the position of the sentence contains the coreferent target. The tokenized *(frequency)* is the *(number)* of token indices 18 and 24, and the token is used as the model input's representative token for the model input. The *(number)*s all refer to the same term but to different things.

Each *(number)* in this case, is weighted with a sentence score of *[CLS]* tokens (no coreference resolution is present in the sentence) and *Sent 1*.

The two candidate *(number)* mentions are semantically distinct, and since *[CLS]* is not coreferenced, the candidate's sentence classification score is calculated as a higher score for *[CLS]*.

Accordingly, the model appears to have prevented *(number)* from being classified as the same entity, and demonstrating the usefulness of the method modeled in a hierarchical structure is useful in resolving coreferences.

However, the model failed to perform cross-reference resolution for *(Tacoma)* (*(Ta)* corresponding to token indices 1 and 8 in the example is input to the model as a representative token), and even in sentence classification, *[CLS]* token misclassified as.

7 | CONCLUSIONS

We created a model that uses head-final characteristics to resolve Korean coreferences. We used hierarchical attention to reflect the sentence context and performed sentence classification for improved understanding of the contextual information of documents composed of

multiple sentences because coreference resolution is processed in units of documents. We used MTL to train the coreference resolution and sentence classification tasks simultaneously, and then conducted various experiments. Furthermore, the model performance was improved using PLMs like BERT and ELECTRA that were pretrained on large-capacity data with the Transformer. Our model achieved a CoNLL F1 of 66.5% on the ETRI test dataset for coreference resolution. The proposed method outperformed the other methods studied.

However, we do have some limitations. In Korean, coreference resolution task still lacks data. This is because the domain of the data we used was limited to QA, it is difficult to apply to various domains. Transformer-based PLM has a deep layer and many attention operations, so the model is heavy and the speed is slow. Also, Generally, when the ensemble method is applied to a system, the speed of the system is proportionally slowed down by the number of models used when inference.

We intend to conduct research on these limitations in the future. We will construct more datasets by handcrafting and translate the CoNLL dataset, and we will improve the quality of the Korean coreference resolution using variational inference to generalize ambiguous words in our model. As PLMs have a large model parameter size, a significant amount of computation is required, and the speed efficiency is poor. We intend to investigate PLM knowledge distillation or develop a PLM with a small architecture to address this issue.

CONFLICT OF INTEREST

The author declares that there are no conflicts of interest.

ORCID

Cheoneum Park https://orcid.org/0000-0001-5386-0483

REFERENCES

1. V. Ng and C. Cardie, *Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution*, (Proceedings of the 19th International Conference on Computational Linguistics, Stroudsburg, PA, USA), 2002. https://doi.org/10.3115/1072228.1072367
2. W. M. Soon, H. T. Ng, and D. C. Y. Lim, *A machine learning approach to coreference resolution of noun phrases*, Comput. Ling. **27** (2001), no. 4, 521–544.
3. K. Lee, L. He, M. Lewis, and L. Zettlemoyer, *End-to-end neural coreference resolution*, (Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark), 2017, pp. 188–197. https://doi.org/10.18653/v1/D17-1018
4. K. Lee, L. He, and L. Zettlemoyer, *Higher-order coreference resolution with coarse-to-fine inference*, (Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, volume 2 (short papers), New Orleans, LA, USA), 2018, pp. 687–692. https://doi.org/10.18653/v1/N18-2108
5. D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, (3rd International Conference on Learning Representations), 2015. https://doi.org/10.48550/arXiv.1409.0473
6. O. Vinyals, M. Fortunato, and N. Jaitly, Pointer networks, in *Advances in neural information processing systems*, 2015, pp. 2692–2700.
7. K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, *Learning phrase representations using RNN encoder–decoder for statistical machine translation*, (Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar), 2014, pp. 1724–1734. https://doi.org/10.3115/v1/D14-1179
8. R. Caruana, *Multitask learning*, *Learning to learn*, S. Thrun and L. Y. Pratt, (eds.), Springer, 1998, pp. 95–133. https://doi.org/10.1007/978-1-4615-5529-2_5
9. S. Ruder, *An overview of multi-task learning in deep neural networks*, arXiv preprint, 2017. https://doi.org/10.48550/arXiv.1706.05098
10. H. Lee, A. Chang, Y. Peirsman, N. Chambers, M. Surdeanu, and D. Jurafsky, *Deterministic coreference resolution based on entity-centric, precision-ranked rules*, Comput. Ling. **39** (2013), no. 4, 885–916.
11. C. Park, K.-H. Choi, C. Lee, and S. Lim, *Korean coreference resolution with guided mention pair model using deep learning*, ETRI J. **38** (2016), no. 6, 1207–1217.
12. M. A. Ur Rahman and V. Ng, *Supervised models for coreference resolution*, (Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore), 2009, pp. 968–977. https://www.aclweb.org/anthology/D09-1101/
13. C. Lee, S. Jung, and C.-E. Park, *Anaphora resolution with pointer networks*, Pattern Recognit. Lett. **95** (2017), 1–7. https://doi.org/10.1016/j.patrec.2017.05.015
14. K. Clark and C. D. Manning, *Improving coreference resolution by learning entity-level distributed representations*, (Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany) 2016, pp. 643–653. https://doi.org/10.18653/v1/P16-1061
15. K. Clark and C. D. Manning, *Deep reinforcement learning for mention-ranking coreference models*, (Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Austin, TX, USA), 2016, pp. 2256–2262. https://doi.org/10.18653/v1/D16-1245
16. J. Li, M.-T. Luong, and D. Jurafsky, *A hierarchical neural autoencoder for paragraphs and documents*, (Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Beijing, China), 2015, pp. 1106–1115. https://doi.org/10.3115/v1/P15-1107
17. I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, *Building end-to-end dialogue systems using generative hierarchical neural network models*, (Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA), 2016, pp. 3776–3784.

18. A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. G. Simonsen, and J.-Y. Nie, *A hierarchical recurrent encoder-decoder for generative context-aware query suggestion*, (Proceedings of the 24th ACM International Conference on Information and Knowledge Management, New York, NY, USA), 2015, pp. 553–562. https://doi.org/10.1145/2806416.2806493
19. R. Lin, S. Liu, M. Yang, M. Li, M. Zhou, and S. Li, *Hierarchical recurrent neural network for document modeling*, (Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal), 2015, pp. 899–907. https://doi.org/10.18653/v1/d15-1106
20. R. Nallapati, B. Zhou, C. N. Dos Santos, C. Gülçehre, and B. Xiang, *Abstractive text summarization using sequence-to-sequence rnns and beyond*, (Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany), 2016, pp. 280–290. https://doi.org/10.18653/v1/k16-1028
21. B. McCann, N. S. Keskar, C. Xiong, and R. Socher, *The natural language decathlon: Multitask learning as question answering*, arXiv preprint, 2018. http://arxiv.org/abs/1806.08730
22. Y. Xu, X. Liu, Y. Shen, J. Liu, and J. Gao, *Multi-task learning with sample re-weighting for machine reading comprehension*, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA), 2019, pp. 2644–2655. https://doi.org/10.18653/v1/n19-1271
23. M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, *Multi-task sequence to sequence learning*, (4th International Conference on Learning Representations), 2016. http://arxiv.org/abs/1511.06114
24. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, (Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA), 2019, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423
25. K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, *ELECTRA: pre-training text encoders as discriminators rather than generators*, (8th International Conference on Learning Representations), 2020. https://openreview.net/forum?id%3Dr1xMH1BtvB
26. W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, *Gated self-matching networks for reading comprehension and question answering*, (Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada), 2017, pp. 189–198.
27. T. Dozat and C. D. Manning, Deep biaffine attention for neural dependency parsing, arXiv preprint, 2016. https://doi.org/10.48550/arXiv.1611.01734
28. T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi, *Simple recurrent units for highly parallelizable recurrence*, (Proceedings of the 2018 Conference on Eempirical Methods in Natural Language Processing, Brussels, Belgium), 2018, pp. 4470–4481. https://doi.org/10.18653/v1/D18-1477
29. D.-A. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (ELU s)*, arXiv preprint, 2015. https://doi.org/10.48550/arXiv.1511.07289
30. G. E. Nasr, E. A. Badr, and C. Joun, *Cross entropy error function in neural networks: Forecasting gasoline demand*, in Proceedings of the fifteenth international florida artificial intelligence research society conference, AAAI Press, 2002, pp. 381–384. http://dl.acm.org/citation.cfm?id%3D646815.708603
31. C. Park, C. Lee, J. Ryu, and H. Kim, *Contextualized embedding- and character embedding-based pointer network for Korean coreference resolution*, In Proceedings of the 30th Annual Conference on Human and Cognitive Language Technology, 2018, pp. 239–242.
32. M. Joshi, O. Levy, L. Zettlemoyer, and D. Weld, *BERT for coreference resolution: Baselines and analysis*, (Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China), 2019, pp. 5803–5808. https://doi.org/10.18653/v1/D19-1588
33. C. Park, J. Shin, S. Park, J. Lim, and C. Lee, *Fast end-to-end coreference resolution for Korean*, (Findings of the Association for Computational Linguistics: EMNLP, Online), 2020, pp. 2610–2624. https://doi.org/10.18653/v1/2020.findings-emnlp.237
34. C. Park, J. Lim, J. Ryu, H. Kim, and C. Lee, *Simple and effective neural coreference resolution for korean language*, ERI J. **43**, (2021), 1038–1048. https://doi.org/10.4218/etrij.2020-0282
35. D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, arXiv preprint, 2014. https://doi.org/10.48550/arXiv.1412.6980

AUTHOR BIOGRAPHY

Cheoneum Park received his BS, MS, and PhD degrees in Computer Science from Kangwon National University, Chuncheon, Republic of Korea, in 2014, 2016, and 2020, respectively. He is now a senior research engineer at AIRS Company of Hyundai Motor Group, Seoul, Republic of Korea. His research interests include natural language processing, natural language understanding, question answering, and deep learning.

How to cite this article: C. Park, *Multi-task learning with contextual hierarchical attention for Korean coreference resolution*, ETRI Journal **45** (2023), 93–104. https://doi.org/10.4218/etrij.2021-0293