Received: 4 August 2023 | Revised: 29 November 2023 | Accepted: 20 December 2023

DOI: 10.4218/etrij.2023-0308

SPECIAL ISSUE

ETRI Journal WILEY

CR-M-SpanBERT: Multiple embedding-based DNN coreference resolution using self-attention SpanBERT

Joon-young Jung

Superintelligence Creative Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea

Correspondence
Joon-young Jung, Superintelligence Creative Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea.
Email: jyjung21@etri.re.kr

Funding information
This study was supported by an Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean Government (23ZS1100, Core Technology Research for Self-Improving Integrated Artificial Intelligence System).

Abstract

This study introduces CR-M-SpanBERT, a coreference resolution (CR) model that utilizes multiple embedding-based span bidirectional encoder representations from transformers, for antecedent recognition in natural language (NL) text. Information extraction studies aimed to extract knowledge from NL text autonomously and cost-effectively. However, the extracted information may not represent knowledge accurately owing to the presence of ambiguous entities. Therefore, we propose a CR model that identifies mentions referring to the same entity in NL text. In the case of CR, it is necessary to understand both the syntax and semantics of the NL text simultaneously. Therefore, multiple embeddings are generated for CR, which can include syntactic and semantic information for each word. We evaluate the effectiveness of CR-M-SpanBERT by comparing it to a model that uses SpanBERT as the language model in CR studies. The results demonstrate that our proposed deep neural network model achieves high-recognition accuracy for extracting antecedents from NL text. Additionally, it requires fewer epochs to achieve an average F1 accuracy greater than 75% compared with the conventional SpanBERT approach.

KEYWORDS
coreference resolution, dependency parsing, multiple embedding

1 | INTRODUCTION

Knowledge on the Internet has been accumulated in large-scale knowledge bases (KBs), such as Yago [1], Wikidata [2], and DBpedia [3], and existing KBs should be expanded to include new knowledge. However, large-scale KBs require considerable effort to expand new knowledge manually. Therefore, studies have been conducted to extract autonomously subject–predicate–object (SPO) tuples from natural language (NL) text to add new knowledge to KBs at a low cost [4]. However, NL sentences may contain ambiguous entities, such as pronouns; accordingly, the SPO tuples extracted from these NL sentences cannot be expressed as knowledge because they may contain these unclear entities. For example, an SPO tuple (he–elected–president) may be extracted from the NL sentence "And he was elected President of the United States 48 years later" in the NL paragraph "Obama was born in Hawaii in 1961. And he was elected President of the United States 48 years later." However, the SPO tuple (he-elected-president) is not knowledgeable. Therefore, for the SPO tuple (he–elected–president)

to become knowledge, it should be converted into an SPO tuple (Obama–elected–president) using the previous sentence "Obama was born in Hawaii in 1961." To extract meaningful knowledge from an NL sentence, an ambiguous entity that may be included in the extracted SPO tuple must be converted to a clear entity. To solve this problem, coreference resolution (CR) methods have been studied. CR is the task of determining all mentions that refer to the same entity in an NL text. For instance, a cluster of mentions referring to the same entity (Obama and he) can be generated from the text "Obama was born in Hawaii in 1961. And he was elected President of the United States 48 years later." Therefore, CR is an important problem in natural language processing (NLP) tasks such as information extraction [4], machine translation [5], and question answering [6]. We propose CR using a multiple embedding-based span bidirectional encoder representation from a transformer (CR-M-SpanBERT) model. The proposed model uses multiple embeddings that integrate NL and dependency relation (DR) embeddings as the input to a deep neural network (DNN) to improve the accuracy of CR.

The remainder of this paper is organized as follows: Section 2 describes related works. Section 3 describes the CR-M-SpanBERT model for an end-to-end CR. Section 4 presents experimental results. Finally, Section 5 presents the concluding remarks.

2 | RELATED WORKS

CR, which recognizes mentions representing the same entity in NL text or dialog, is an important and challenging problem in NLP. Therefore, research on CR has been extensively conducted in NLP.

The mention-pair model classifies whether two mentions of the NL text are coreferent [7, 8]. Soon and others [9] proposed a learning-based CR model based on decision-tree induction for noun phrase (NP) mentions to present potential mention pairs. Ng and others [10] proposed the best-first clustering model, which selects the NP with the highest cross-referencing probability from among preceding NPs as an antecedent. Bengtson and Roth [11] proposed a simple but effective CR model based on a pairwise model and strong set of features. However, this paired model has certain limitations. A transitivity problem may occur because each candidate antecedent for an anaphora is estimated independently without considering the others. The problem of expressiveness can arise when the pairwise features of two mentions alone are insufficient to make a coreference decision. For example, "Mr. Obama" and "Obama" are coreferent; however, the model may incorrectly determine that "Obama" and "she" are coreferent.

The entity-mention model overcomes the limitations of the mention-pair model. The entity-mention model determines whether an NP mention is consistent with the preceding cluster. Björkelund and Kuhn [12] proposed a structured perceptron model for CR using nonlocal features and beam search. Clark and Manning [13] proposed an entity-centric coreference model to build incrementally coreference chains using entity-level features between clusters of mentions. Wiseman and others [14] proposed an end-to-end coreference model that uses recurrent neural networks to represent entity clusters directly from mentions.

Although the entity-mention model addresses the weaknesses of the mention-pair model, it does not identify the most probable antecedents. However, the mention-ranking model can determine the most probable candidate antecedents of NP mentions. The mention-ranking model simultaneously ranks all candidate antecedents of a mention and directly selects the antecedent with the highest score as the most probable candidate antecedent. Durrett and Klein [15] proposed a CR model using a simple and homogeneous feature set to achieve high performance. Wiseman and others [16] proposed a simple nonlinear coreference model for learning intermediate feature representations for anaphoricity detection and antecedent ranking using raw nonconjoined features. Marasovic and others [17] proposed a mention-ranking model that uses a long short-term memory (LSTM) Siamese network to learn how NP mentions relate to their antecedents.

Recently, an end-to-end model was proposed that performs both mention detection and coreference prediction. Lee and others [18] proposed an end-to-end model using bidirectional LSTM that considers all spans of a document as potential mentions. Kantor and Globerson [19] proposed an end-to-end model based on bidirectional encoder representations from transformer (BERT) embeddings that use entity equalization to represent each mention of a cluster through every mention of the cluster. Joshi and others [20] proposed an end-to-end model using BERT that considers all spans of a document as potential mentions. Subsequently, Joshi and others [21] proposed a model using an extended BERT (SpanBERT) to mask spans and train span boundary representations. Park and others [22] proposed an end-to-end model using BERT for CR. Kirstain and others [23] proposed a CR model using the Longformer, as proposed by Beltagy and others [24], without span representations for simple but efficient coreferencing. To improve the performance of CR using an end-to-end DNN, we propose the CR-M-SpanBERT model.

3 | CR-M-SPANBERT

Various language models (LMs), such as LSTM [18], BERT [19], SpanBERT [21], T-zero (T0) [25], and multilingual pretrained text-to-text transfer transformer (mT5) [26], have been utilized to enhance the performance of CR. In CR, mention clusters in NL texts are recognized using the output embeddings of the LM. In this study, the CR was performed using SpanBERT as the LM. The use of a larger LM parameter size correlates with better average F1 performance (average F1 of the MUC, B^3, and $CEAF_{\varphi 4}$ metrics). For the LMs, LSTM, BERT, SpanBERT, T0, and mT5, with respective parameter sizes of 15 M, 340 M, 340 M, 3 B, and 13 B, exhibited average F1 performances of 73.0, 76.9, 79.6, 82.3, and 83.3, respectively. While the mT5 and T0 models exhibited better average F1 performances than SpanBERT, their parameter sizes were approximately 38 and 9 times larger than that of SpanBERT, respectively. Therefore, employing mT5 and T0 as LM for the CR incurs a considerable computational cost.

BERT, which uses a self-supervised training method that masks individual words or subwords in NL text, exhibits good performance in the field of NLP [27]. When masking some of the wordpiece tokens in a masked language model, BERT independently selects each token from a sequence of words or subword tokens at random. However, because the CR task recognizes the relationships between spans in an NL text, it must better represent and predict the spans in the NL text. SpanBERT, proposed by Joshi and others [21], is a self-supervised learning model that randomly selects contiguous spans that exhibit better performance than BERT for CR.

Figure 1 shows the CR training process using the CR-M-SpanBERT model. This involves extracting sentences from the CR data and generating dependency trees and DR sentences. NL and DR sentences were then subjected to NL and DR embedding to generate multiple embeddings. The generated multiple embeddings were input into the SpanBERT model, which subsequently performed span representation and antecedent classification. The CR model was trained using a loss function.

The proposed DNN model for CR consists of multiple embeddings, a self-attention SpanBERT, attention-based span representation, and antecedent classification. Multiple embeddings parse the dependencies of NL and then integrate the NL and DR embeddings. Self-attention SpanBERT generates an output sequence that can be used for span representation using multiple embeddings as the input sequence. The attention-based span representation embeds a span using the self-attention SpanBERT outputs. Antecedent classification detects mentions using attention-based span representation output and determines the antecedent of an anaphora among the mentions. The details of each module follow.

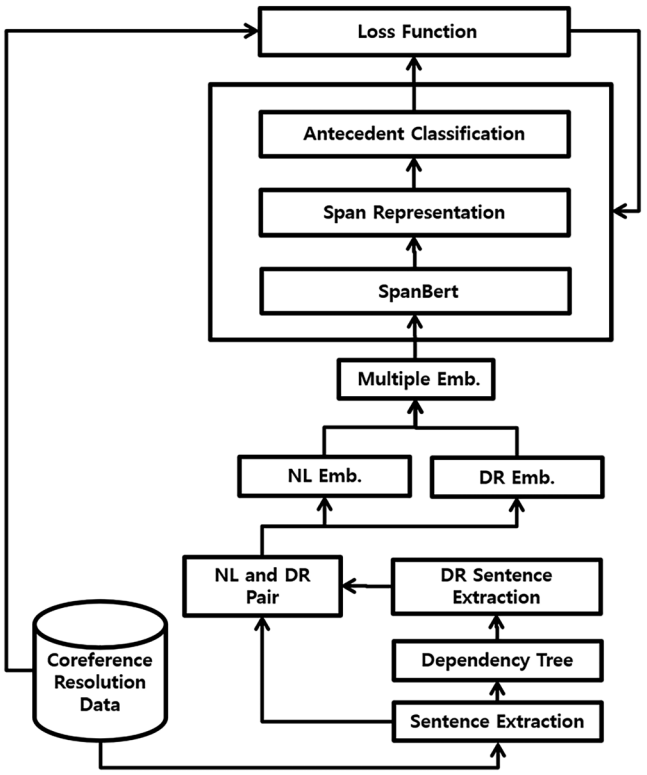


FIGURE 1 Coreference resolution model using a multiple embedding-based span bidirectional encoder representations from transformers (CR-M-SpanBERT).

3.1 | Multiple embeddings

To perform NLP, each word of the NL text must be embedded, and word-embedding techniques such as semantic-based c-gram [28], skip-gram [29], and glove [30] have been proposed. However, because each word in an NL text contains semantic and syntactic information, word embeddings that include this information in a complex manner must be performed to understand the NL text more accurately.

In the case of CR, it is necessary to understand the syntax and semantics of NL texts simultaneously. Therefore, multiple embeddings were performed that included syntactic and semantic information for each word in the NL text. To analyze the syntactic information of the NL text and establish dependency relationships between the words in a sentence, a parse tree was generated using a dependency parser. Specifically, the parse tree for the sentence "Google has announced their Android Platform for mobile devices" is shown in Figure 2. The syntactic

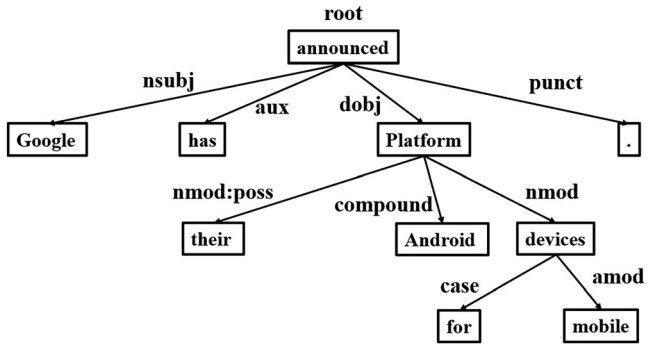


FIGURE 2 Dependency tree in a natural language (NL) sentence.

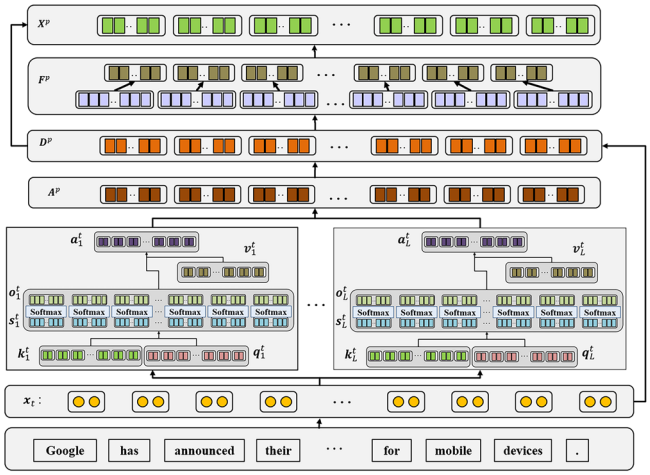


FIGURE 3 M-SpanBERT.

structures of sentences are determined using the fast and accurate dependency parser proposed by Chen and others [31].

Only a few types of DRs exist between the words in sentences created using a dependency parser [32]. Therefore, the syntactic embeddings of DRs are generated using a simple and high-performance skip-gram. In other words, DRs such as nsubj, aux, and amod generated by a dependency parser are trained and transformed into embedding vectors using the skip-gram model, which is the Word2vec model proposed by Mikolov and others [29]. Semantic embedding separates NL text into words and performs embedding for each word. That is, words in a sentence, such as "mobile" and "devices," are transformed into embedding vectors using the Word2vec model. Positional embedding was performed to generate relative position information for tokens in NL texts using a sinusoidal function [33].

By integrating semantic, syntactic, and positional embeddings, as shown in (1), multiple embeddings, including DRs and semantic information, were generated for each token in the NL text. The multiple embedding-based SpanBERT (M-SpanBERT), as depicted in Figure 3, generates an output sequence that can be used for span representation using multiple embeddings.

$$\boldsymbol{x}_t = \Phi_m(\boldsymbol{s}_t, \boldsymbol{r}_t, \boldsymbol{p}_t) \tag{1}$$

where $\boldsymbol{x}_t$ denotes the multiple embeddings for the t^{th} token in NL text; $\boldsymbol{s}_t$, $\boldsymbol{r}_t$, and $\boldsymbol{p}_t$ are the semantic, syntactic, and positional embeddings for the t^{th} token in NL text, respectively; and Φ_m is the element-wise addition and concatenation function for multiple embeddings.

3.2 | Self-attention SpanBERT

The self-attention value was generated using a multiple embedding sequence, as shown in Figure 3. The key (k_i^t), query (q_i^t), and value (v_i^t) corresponding to the t^{th} token of the i^{th} attention were generated by applying a feedforward neural network (FFNN) to each token of multiple embeddings (x_t). An attention score (s_i^t) was generated for each token using the key and query. The softmax value (o_i^t) of the attention score is

$$o_i^{t,l} = \frac{\exp\left(s_i^{t,l}\right)}{\sum_{j=1}^{T} \exp\left(s_i^{t,j}\right)}, \tag{2}$$

where $o_i^{t,l}$ is the softmax value of the t^{th} token using the l^{th} token of the i^{th} attention, $s_i^{t,l}$ is the attention score of the t^{th} token with the l^{th} token of the i^{th} attention, and T is the sequence length.

The self-attention value (a_i^t) is given by

$$\boldsymbol{a}_i^t = o_i^t \bullet \boldsymbol{V}_i, \tag{3}$$

where $\boldsymbol{a}_i^t$ is the self-attention value of the t^{th} token of the i^{th} attention, o_i^t is the softmax value of the t^{th} token of the i^{th} attention, and $\boldsymbol{V}_i$ is the value of all tokens of the i^{th} attention. The matrix multiplication operator is expressed as $\bullet$.

The multiple self-attention value concatenates all attention values as

$$\boldsymbol{A}^p = \Phi_c(\boldsymbol{A}_1, ..., \boldsymbol{A}_L)\boldsymbol{W}^p + \boldsymbol{b}^p, \tag{4}$$

where $\boldsymbol{A}^p$ is the multiple self-attention value in the p^{th} layer, $\boldsymbol{A}_1$ is the first attention value of all tokens, $\boldsymbol{A}_L$ is the last attention value of all tokens, Φ_c is the concatenation function, and $\boldsymbol{W}^p$ and $\boldsymbol{b}^p$ are the weight matrix and bias in the p^{th} layer, respectively.

Decoding in the p^{th} layer is performed by applying the output of the previous layer, the $(p-1)^{\text{th}}$ layer, to the multiple self-attention values in the p^{th} layer as follows:

$$\boldsymbol{D}^p = \Phi_d\left(\boldsymbol{A}^p, \boldsymbol{X}^{p-1}\right), \tag{5}$$

where D^p is the decoding value in the p^{th} layer, $\boldsymbol{X}^{p-1}$ is the output of the $(p\text{-}1)^{\text{th}}$ layer, $\boldsymbol{X}^0$ is a multiple embedding sequence, and Φ_d is the residual connection and normalization function.

An FFNN is constructed using decoding values as follows:

$$\boldsymbol{f}_n^p = \sigma_n\left(\boldsymbol{f}_{n-1}^p \boldsymbol{W}_n^p + \boldsymbol{b}_n\right), \tag{6}$$

where $\boldsymbol{f}_0^p = \boldsymbol{D}^p$, σ_n is the activation function of the n^{th} layer of the FFNN, n is the FFNN layer (with values from 1 to N), $\boldsymbol{W}_n^p$ is a weight matrix mapping the $(n\text{-}1)^{\text{th}}$ layer to the n^{th} layer of the FFNN, and $\boldsymbol{b}_n$ is the bias of the n^{th} layer of the FFNN. The dimensions of $\boldsymbol{f}_N^p$ were the same as those of $\boldsymbol{D}^p$.

The output of the p^{th} layer, obtained from the decoding value and FFNN result, is as follows:

$$\boldsymbol{X}^p = \Phi_d(\boldsymbol{F}^p, \boldsymbol{D}^p), \tag{7}$$

where $\boldsymbol{X}^p$ is the output vector of the p^{th} layer and $\boldsymbol{F}^p$ (i.e., $\boldsymbol{f}_N^p$) is the FFNN result of the p^{th} layer.

M-SpanBERT utilizes the existing SpanBERT, and fine-tuning is performed with span representation and antecedent classification, as depicted in Figure 1.

3.3 | Attention-based span representation

The output sequence (X^L) is the output vector of the last layer of the M-SpanBERT. For the CR, a span representation was performed using this output sequence. The span indices (e_t) consist of candidate spans with the maximum span width. The masked span indices ($\bar{e}_t$) are masked by the maximum span width as follows:

$$m_{t,i} = \begin{cases} 0 & (l_{t,i} \le \text{max span width}) \\ -\text{inf} & (l_{t,i} \ge \text{max span width}) \\ -\text{inf} & (i(e_{t,i}) \ge \text{len(sentence)}) \end{cases}, \qquad \bar{e}_t = e_t \bigodot m_t, \tag{8}$$

where $m_{t,i}$ is the attention mask of the i^{th} word in the t^{th} candidate span, $l_{t,i}$ is the difference between the index values of the start and i^{th} words in the t^{th} candidate span, $i(e_{t,i})$ is the i^{th} word index value of the t^{th} candidate span, m_t is the mask vector of the t^{th} candidate span, e_t is the span index vector of the t^{th} candidate span, and $\bar{e}_t$ is the masked span index of the t^{th} candidate span. Element-wise addition is denoted by $\bigodot$.

The process of generating span-attention embeddings is shown in Figure 4.

The M-SpanBERT attention score (s_i) for each token was obtained using the M-SpanBERT output sequence and FFNN, as follows:

$$s_i = \sigma_s\left(\sum_{t=1}^{T} \boldsymbol{x}_{i,t}^L \boldsymbol{w}_t + \boldsymbol{b}_s\right), \tag{9}$$

where s_i is the i^{th} M-SpanBERT attention score in the M-SpanBERT output sequence, σ_s is the activation function of the FFNN, T is the size of the M-SpanBERT embedding vector, $\boldsymbol{x}_{i,t}^L$ is the t^{th} value of the i^{th} M-SpanBERT embedding vector in the M-SpanBERT output sequence ($\boldsymbol{X}^L$), $\boldsymbol{w}_t$ is the weight matrix mapping the t^{th} value of the M-SpanBERT embedding to the M-SpanBERT attention score, and $\boldsymbol{b}_s$ is the FFNN bias.

The M-SpanBERT attention score corresponding to the index of each word was applied to the masked span indices ($\bar{e}_t$) to create score span indices ($\dot{e}_t$). Subsequently, softmax was applied to each row of the score span indices for the attention span indices ($\bar{\bar{e}}_{t,i}$), as shown in (10),

$$\bar{\bar{e}}_{t,i} = \frac{\exp(\dot{e}_{t,i})}{\sum_{j=1}^{W} \exp(\dot{e}_{t,j})}, \tag{10}$$

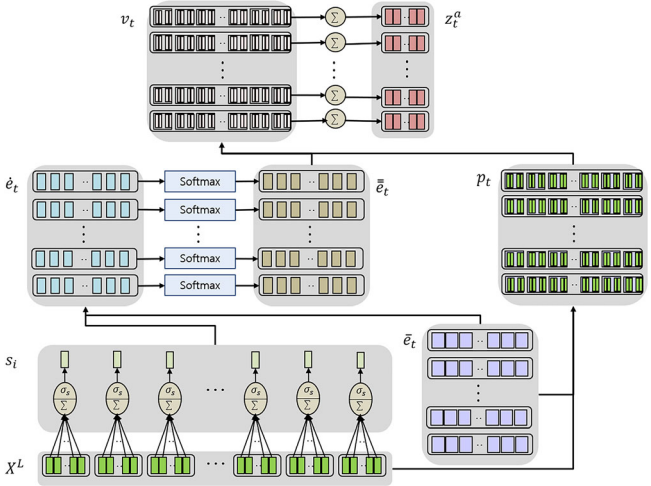


FIGURE 4 Process used to generate span-attention embeddings using the output sequence and masked span index.

where $\bar{\bar{e}}_{t,i}$ is the softmax value of $\dot{e}_{t,i}$; $\dot{e}_{t,i}$ is the i^{th} word of the t^{th} candidate span in the score-span indices, and W is the maximum span width.

The M-SpanBERT output embedding (X^L) corresponding to the index of each word is applied to the span indices (e_t) to create the span embedding (p_t).

Span-attention embedding (z_i^a) is created using the M-SpanBERT span embedding and attention span indices, as shown in (11) and (12). To create the M-SpanBERT value embedding, the M-SpanBERT span embedding was multiplied by the attention span indices, as shown in (11). Span-attention embedding was created by the element-wise addition of the candidate span vectors of the M-SpanBERT value embedding, as expressed by (12).

$$v_{t,i} = p_{t,i} \bullet \bar{\bar{e}}_{t,i}, \tag{11}$$

$$z_t^a = \sum_{j=1}^{W} v_{t,j}, \tag{12}$$

where $v_{t,i}$ is the i^{th} embedding of the t^{th} candidate span in the M-SpanBERT value embedding, $p_{t,i}$ is the i^{th} embedding of the t^{th} candidate span in the M-SpanBERT span embedding, symbol $\bullet$ is a multiplication operator, and z_t^a is the t^{th} candidate span in the span-attention embedding.

The attention-based span representation (z) concatenates the span start, end, width, and attention embeddings as follows:

$$z_t = \Phi_c\left(z_t^s, z_t^e, z_t^w, z_t^a\right), \tag{13}$$

where z_t is the t^{th} attention-based span representation and z_t^s, z_t^e, z_t^w, and z_t^a are the span's start, end, width, and attention embeddings of the t^{th} span, respectively.

3.4 | Attention-based antecedent classification

The procedure for generating pair embeddings of anaphora and antecedent for antecedent classification is shown in Figure 5.

Mention detection is the detection of span representations observed as mentions in NL texts. First, the mention score for each span representation was obtained. The span representations with high-mention scores were then identified. The mention score for each span representation was determined using the FFNN [see (14)]. The input data of the FFNN layer are a span representation (z).

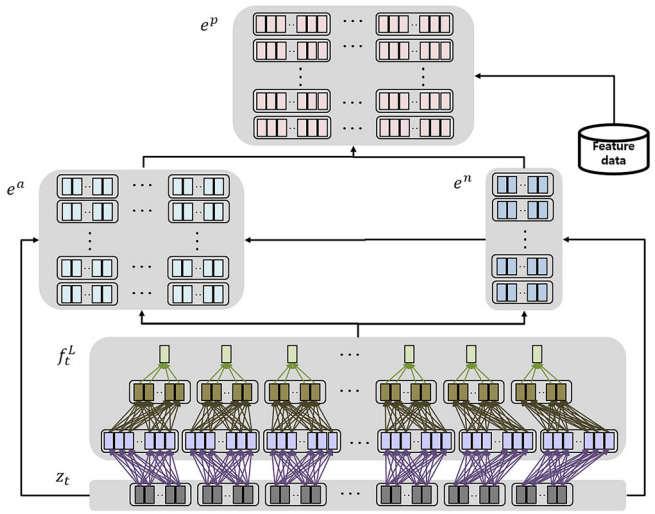


FIGURE 5 Pair embeddings of anaphora and antecedent.

$$\boldsymbol{f}_t^n = \sigma^n\left(\boldsymbol{f}_t^{n-1}\boldsymbol{w}^n + \boldsymbol{b}^n\right), \tag{14}$$

where $\boldsymbol{f}_t^n$ is the t^{th} hidden vector of the n^{th} layer of the FFNN ($\boldsymbol{f}_t^0 = \boldsymbol{z}_t$), σ^n is the activation function of the n^{th} layer, n is the FFNN layer (from 1 to L), $\boldsymbol{w}^n$ is the weight matrix mapping the $(n\text{-}1)^{\text{th}}$ layer to the n^{th} layer, and $\boldsymbol{b}^n$ is the bias of the n^{th} layer.

In the last layer of the FFNN for the mention score, the output of each span (f_t^L) has a mention score. Using these mention scores (f_t^L, where t ranges from 1 to T, and T is the number of span representations), mentions were detected by selecting spans with high-mention scores. Approximately 40% of the spans were selected for mention. A detected mention is a subset of span representations.

The antecedent in the CR refers to the anaphora in the NL text. Using the mention score, several antecedent scores for each anaphora were measured using FFNN.

Anaphora embeddings consist of span representations with high-mention scores as follows:

$$e^n = \left[z^h, z^{h-1}, ..., z^{h-k-1}\right], \tag{15}$$

where e^n is the anaphora embedding, z^h is the span representation with the highest mention score, and z^{h-k-1} is the span representation with the k^{th} largest mention score.

Antecedent embeddings consist of span representations with high-mention score for each anaphora as follows:

$$e_k^a = \left[z_k^h, z_k^{h-1}, ..., z_k^{h-c-1}\right], \tag{16}$$

where e_k^a is the antecedent embedding of the k^{th} anaphora; z_k^h is the span representation with the highest

mention score for the k^{th} anaphora, and z_k^{h-c-1} is the span representation with the c^{th} largest mention score for the k^{th} anaphora.

Pair embedding is generated using data such as antecedent embedding, as shown in (17). Similarity, embedding is represented by the element-wise multiplication of the anaphora and antecedent embeddings. Feature embedding represents features such as speaker identification (ID) and antecedent distance between the anaphora and candidate antecedents.

$$e_{k,i}^{p} = \Phi_c\left(e_k^n, e_{k,i}^a, e_{k,i}^s, e_{k,i}^f\right), \tag{17}$$

where $e_{k,i}^p$, e_k^n, $e_{k,i}^a$, $e_{k,i}^s$, and $e_{k,i}^f$ are the pair, anaphora, antecedent, similarity, and feature embeddings of the k^{th} anaphora and i^{th} antecedent, respectively.

The procedure for generating antecedent scores and attention-based mention embeddings using pairs of anaphoras and antecedent embeddings is shown in Figure 6.

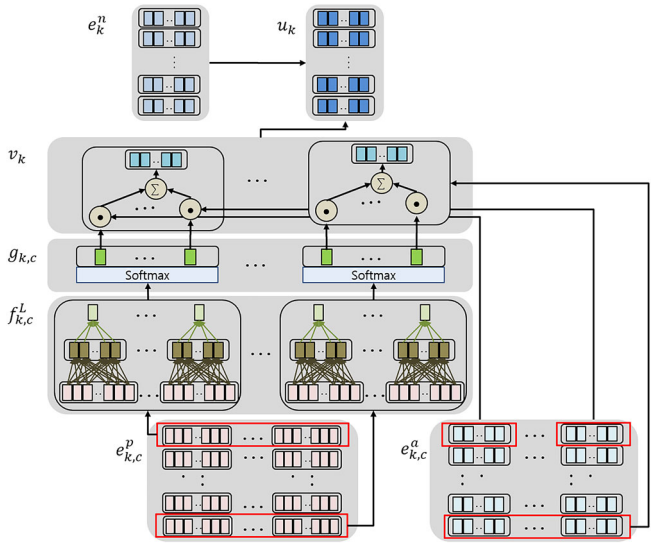


FIGURE 6 Antecedent scores and attention-based mention embeddings.

The antecedent score was measured using the generated pair embedding as the input data of the FFNN, as follows:

$$f_{k,c}^n = \sigma^n\left(\boldsymbol{f}_{k,c}^{n-1}\boldsymbol{w}^n + \boldsymbol{b}^n\right), \tag{18}$$

where $f_{k,c}^n$ are the hidden vectors of the k^{th} anaphora and c^{th} candidate antecedent in the n^{th} layer of the FFNN ($f_{k,c}^0 = e_{k,c}^p$), respectively.

In the last layer of the FFNN for the antecedent score, the output of the pair between the anaphora and antecedent ($f_{k,c}^L$) has an antecedent score. These antecedent scores were used as attention scores.

The softmax value ($g_{k,c}$) of the c^{th} antecedent of the k^{th} anaphora was calculated using the candidate antecedent scores of the k^{th} anaphora, as shown in(19). The attention value of the k^{th} anaphora (v_k) is the product of the antecedent embedding and softmax values after matrix multiplication, as expressed in (20). Attention-based mention embedding was generated by applying mention embedding to the attention values, as shown in (21).

$$g_{k,c} = \frac{\exp\left(f_{k,c}^L\right)}{\sum_{j=1}^{C}\exp\left(f_{k,j}^L\right)}, \tag{19}$$

$$v_k = \sum_{j=1}^{C}\left(e_{k,j}^a \bullet g_{k,j}\right), \tag{20}$$

$$u_k = y * v_k + (1-y) * e_k^n, \tag{21}$$

where $g_{k,c}$ is the softmax value of the antecedent score ($f_{k,c}^L$) of the c^{th} antecedent of the k^{th} anaphora, C is the number of candidate antecedents of an anaphora, v_k is the attention value of the k^{th} anaphora, $e_{k,j}^a$ is the antecedent embedding of the j^{th} antecedent of the k^{th} anaphora, the symbol $\bullet$ denotes matrix multiplication, u_k is the k^{th} attention-based mention embedding (k ranges from 1 to K, where K is the number of anaphors), and y is a scalar value which ranges between 0 and 1.

TABLE 1 Performance on OntoNotes coreference resolution (CR) benchmark.

	B^3			$CEAF_{\varphi 4}$			MUC			
Model	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**Average F1**
Lee 4.et al. [18]	72.2	69.5	70.8	68.2	67.1	67.6	81.4	79.5	80.4	73.0
Kantor and Globerson [19]	73.3	76.2	74.7	72.4	71.1	71.8	82.6	84.1	83.4	76.6
BERT [20]	76.5	74.0	75.3	74.1	69.8	71.9	84.7	82.4	83.5	76.9
SpanBERT [21]	78.3	77.9	78.1	76.4	74.2	75.3	85.8	84.8	85.3	79.6
CR-M-SpanBERT (ours)	81.4	78.0	79.6	77.9	75.1	76.5	87.0	84.2	85.6	80.6

Attention-based mention embeddings are used to recognize attention-based antecedent scores using (16)–(18) recursively. Consequently, an attention-based antecedent score $\left(\bar{f}_{k,c}^{L}\right)$ was obtained.

Supervised learning was performed so that the antecedent of an anaphora had the highest antecedent score. The mask of the antecedent score ($m_{k,c}$) has a value of zero if the cluster identity (ID) of anaphora (k) and the cluster ID of the candidate antecedent of anaphora (c) are the same; otherwise, it has a value of -inf, as shown in (22). The masked antecedent score ($r_{k,c}$) only maintains the antecedent score value that matches the cluster ID of the anaphora and the antecedent in the antecedent score; the remaining score is -inf, as shown in (23). The loss function (L) for learning is given by (24). Learning was performed such that the loss value was minimized.

$$m_{k,c} = \begin{cases} 0\,(\text{clusterID}(k) = \text{clusterID}(c)) \\ -\inf\,(\text{clusterID}(k) \neq \text{clusterID}(c)) \end{cases}, \tag{22}$$

$$r_{k,c} = \bar{f}_{k,c}^{L} + m_{k,c}, \tag{23}$$

$$L = \sum_{i=1}^{K} \left\{ \log\left(\sum_{j=1}^{C} \exp\left(\bar{f}_{i,j}^{L}\right)\right) - \log\left(\sum_{j=1}^{C} \exp\left(r_{i,j}\right)\right) \right\}, \tag{24}$$

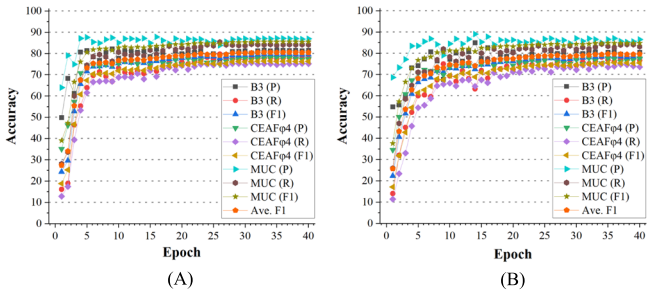


FIGURE 7 Coreference resolution (CR) accuracy test results using the (A) CR-M-SpanBERT and (B) SpanBERT models.

where $m_{k,c}$ is the mask of the antecedent score of the k^{th} anaphora and c^{th} candidate antecedent, $r_{k,c}$ is the masked antecedent score of the k^{th} anaphora and c^{th} candidate antecedent, and L is the loss value.

The antecedent of an anaphora is recognized in an NL text through an attention-based CR system using M-SpanBERT.

EXPERIMENTAL RESULTS

To evaluate the effectiveness of the proposed attention-based CR system using M-SpanBERT, the proposed model was compared with a model that utilizes SpanBERT as the LM in CR studies. The precision, recall, and F1 of the MUC, B^3, and $CEAF_{\varphi 4}$ metrics were used for evaluating the performance of the CR model [34]. Precision and recall are the indicators of the correct proportion of resolved coreference information and fraction of correct coreference information that has been resolved, respectively. F1 measures the balance between recall and precision. Therefore, the average F1 values of the MUC,

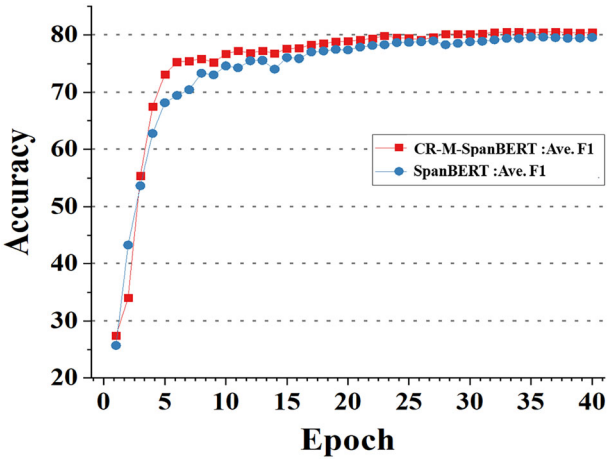


FIGURE 9 Coreference resolution (CR) accuracy comparison between the CR-M-SpanBERT and SpanBERT models using the average (Ave.) F1 scores of the B3, CEAFφ4, and MUC metrics.

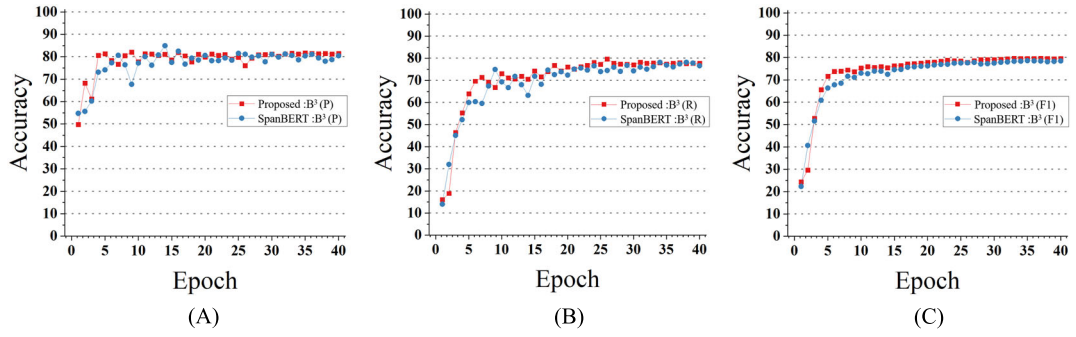


FIGURE 8 Comparison of the coreference resolution (CR) accuracy using the proposed model (CR-M-SpanBERT) and SpanBERT: (A) precision, (B) recall, and (C) F1 score responses of the B^3 metric.

B^3, and $CEAF_{\varphi 4}$ metrics are important indicators for comparing the performances of CR models.

Performance evaluation was conducted using an NVIDIA V-100 (32GB) graphics processing unit, utilizing approximately 31 GiB of memory and requiring approximately 43 min to complete one epoch. The results of the performance on the OntoNotes CR benchmark are listed in Table 1. The proposed CR-M-SpanBERT achieved a higher CR accuracy than the other CR models that employed LSTM, BERT, and SpanBERT as language models. The precision, recall, and F1 of the MUC metric for the proposed model were 86.9, 84.2, and 85.6, respectively. The precision, recall, and F1 of the B^3 metric of the proposed model were 81.2, 78, and 79.6, respectively. The precision, recall, and F1 of the $CEAF_{\varphi 4}$ metric for the proposed model were 77.9, 75.1, and 76.5, respectively. The average F1 values of the MUC, B^3, and $CEAF_{\varphi 4}$ metrics were 80.6. Thus, the proposed model had the highest average F1 value compared with the other CR models that employed LSTM, BERT, and SpanBERT as language models.

Among the existing CR models compared in Table 1, the SpanBERT model proposed by Joshi et al. [21] is the most similar to the CR-M-SpanBERT model proposed in this study because both utilize SpanBERT as the LM. However, this study conducted dependency parsing for NL sentences, generated embeddings for the DR, and created multiple embeddings to incorporate simultaneously the syntactic and semantic information of NL sentences for the CR. In addition, this study performs span and mention representations using attention. The performances of SpanBERT and CR-M-SpanBERT are compared across the metrics in Figures 7–9 and Tables 2 and 3.

Figure 7 shows the evaluation test results obtained using the CR-M-SpanBERT and SpanBERT models. In the CR-M-SpanBERT model, the evaluation test results show that as the number of epochs increases, the CR accuracy improves, as shown in Figure 7A. In the SpanBERT model, the CR accuracy was improved, as shown in Figure 7B. A performance comparison of the B^3 metric between the CR-M-SpanBERT and SpanBERT models is shown in Figure 8. The comparison focused on the precision, recall, and F1 values of the B^3 metric. It was observed that the CR-M-SpanBERT model exhibited fewer fluctuations in precision and recall as a function of the number of epochs, thus achieving higher F1 scores than that of the SpanBERT model.

Table 2 lists the maximum and minimum values of the CR accuracy measured in the evaluation tests using the CR-M-SpanBERT and SpanBERT models. At the same time, the maximum precision values for B^3, $CEAF_{\varphi 4}$, and MUC are better for the SpanBERT model than for the CR-M-SpanBERT model. The evaluation of the instances where maximum precision was measured is

TABLE 2 Minimum (min.) and maximum (max.) accuracies for SpanBERT and CR-M-SpanBERT across all epochs in evaluation tests.

Model	B^3						$CEAF_{\varphi 4}$						MUC						Average F1	
	P		R		F1		P		R		F1		P		R		F1			
	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
SpanBERT	54.8	84.95	14.04	78.23	22.35	78.61	34.55	78.21	11.38	74.95	17.12	75.53	68.62	88.99	25.93	84.31	37.64	84.83	25.70	79.64
CR-M-SpanBERT	49.8	82.07	16.13	79.72	24.37	**79.64**	35.07	77.92	12.87	75.61	18.83	**76.49**	63.97	87.77	28.12	85.35	39.07	**85.56**	27.42	**80.56**

Note: Values in bold emphasis highlight that the maximum F1 scores for B_3, $CEAF_{\varphi 4}$, and MUC are better in the CR-M-SpanBERT model compared with the SpanBERT model.

TABLE 3 Accuracy at the epoch with maximum precision for SpanBERT and CR-M-SpanBERT.

	B^3				$CEAF_{\varphi 4}$				MUC			
Model	**P**	**R**	**F1**	**Epoch**	**P**	**R**	**F1**	**Epoch**	**P**	**R**	**F1**	**Epoch**
SpanBERT	84.95	63.24	72.5	14	78.21	71.12	74.49	26	88.99	74.02	80.82	14
CR-M-SpanBERT	82.07	66.86	73.69	8	77.92	75.11	76.49	37	87.77	79.65	83.51	15

TABLE 4 Comparison of CR performances with large LM models.

Model	**B^3 (F1)**	**$CEAF_{\varphi 4}$ (F1)**	**MUC (F1)**	**Ave. F1**
seq2seq ($mT5_{XXL}$) [26]	82.6	79.5	87.8	83.3
ASP + $T0_{3B}$ [25]	81.5	78.4	86.9	82.3
CR-M-SpanBERT	79.6	76.5	85.6	80.6

Abbreviation: CR, coreference resolution.

TABLE 5 Comparison of CR performances with the use of multiple embeddings.

Model	**B^3 (F1)**	**$CEAF_{\varphi 4}$ (F1)**	**MUC (F1)**	**Ave. F1**
LSTM [18]	70.8	67.6	80.4	73.0
└ M-LSTM	71.5	68.0	80.5	73.3
BERT [20]	75.3	71.9	83.5	76.9
└ M-BERT	76.1	72.9	83.6	77.5
SpanBERT [21]	78.1	75.3	85.3	79.6
└ M-SpanBERT	79.5	76.2	85.5	80.4

Abbreviation: CR, coreference resolution.

provided in Table 3. During the training process, the recall tended to be lower in the epoch in which the SpanBERT model achieved maximum precision. Figure 8 shows that in the CR of the SpanBERT model, there is a considerable fluctuation in the precision and recall values as a function of the epoch, this leading to instances where the maximum precision is higher for the SpanBERT model than for the CR-M-SpanBERT model. Therefore, simultaneously considering false positives and false negatives in the F1 value is more appropriate for performance measurements in CR. In Table 2, bold emphasis highlights that the maximum F1 scores for B^3, $CEAF_{\varphi 4}$, and MUC are better in the CR-M-SpanBERT model compared to the SpanBERT model.

In terms of the maximum F1 scores for B^3, $CEAF_{\varphi 4}$, and MUC, CR-M-SpanBERT outperformed SpanBERT with performance improvements of 1.03, 0.96, and 0.73, respectively. As a result, the average F1 scores of B^3, $CEAF_{\varphi 4}$, and MUC demonstrate a superior performance improvement of 0.92. The performance of CR using the SpanBERT model as the LM yielded an average F1 value of 79.6 [21]. In this study, the average F1 value of 80.6 constitutes the best CR performance using SpanBERT.

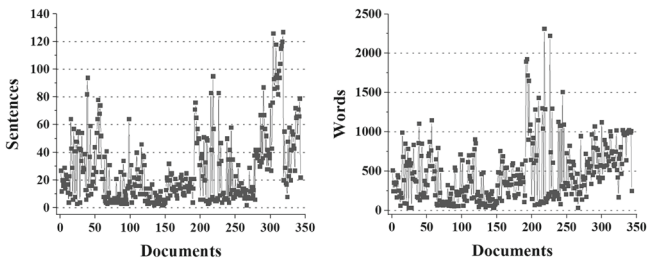
Figure 9 shows the differences in accuracy according to the number of epochs for the average F1 of the B^3, $CEAF_{\varphi 4}$, and MUC metrics for the SpanBERT and the proposed model. The CR accuracy of the proposed model increased to 80.56, whereas that of SpanBERT increased to 79.64 for the average F1 score. The proposed model had an average F1 accuracy greater than 75 at six epochs, whereas SpanBERT achieved the same accuracy at 12 epochs. Therefore, the proposed model learned the CR faster. Furthermore, because the proposed model exhibited lesser fluctuation as the number of epochs increased, the performance improvement as the number of epochs increased was more stable than that of SpanBERT.

As shown in Table 4, Bohnet and others [26] and Liu and others [25] reported average F1 scores of 83.3 and 82.3 for CR, respectively. However, to achieve such a high performance, they used $mT5_{XXL}$ (LM with a parameter size of 13 billion) and $T0_{3B}$ (LM with a parameter size of 3 billion). In comparison, SpanBERT had a parameter size of 340 million. The proposed method employed SpanBERT as the LM and incorporated a skip-gram for multiple embeddings. The parameter size of the skip-gram model was approximately 0.3 million. Therefore,

the LM parameter size of the CR-M-SpanBERT model was nearly the same as that of the SpanBERT model. Miculicich and Henderson [35] reported an average F1 score of 80.5 for the CR based on SpanBERT. The method proposed in this study achieved an average F1 score of 80.6, showing a slightly better performance in CR when SpanBERT was used as the LM.

To investigate the impact of multiple embeddings on CR performance, experiments were conducted by applying multiple embeddings to the LSTM, BERT, and SpanBERT models. As shown in Table 5, the multiple embedding-based LSTM (M-LSTM) model, which incorporates multiple embeddings into the LSTM, exhibited an improved average F1 performance of 73.3. The multiple embedding-based BERT (M-BERT) and M-SpanBERT models demonstrated average F1 performances of 77.5 and 80.4, respectively. The performance of the average F1 in the M-LSTM, M-BERT, and M-SpanBERT models was improved by 0.3, 0.6, and 0.8, respectively, compared with the LSTM, BERT, and SpanBERT models.

Multiple embeddings refer to those that include information regarding DR, where DR encompasses the syntactic information of each word in a sentence. The CR task involved grouping mentions in the NL texts that conveyed the same meaning into the same cluster. As the sentence components of mentions in NL texts are often subjects and objects, incorporating DR enhances the accuracy of CR. For example, a cluster of mentions referring to the same entity (Obama, he) can be generated from the text "Obama was born in Hawaii in 1961. He was elected as the President of the United States 48 years later." Here, "Obama" and "He" are both subjects in their respective sentences.



(A) Number of sentences (B) Number of words

FIGURE 10 Number of sentences and words included in each document of the test data.

The document length characteristics for the English-language data in OntoNotes used for benchmark testing are shown in Figure 10. The test data consisted of 343 documents. The number of sentences in each document ranged from a minimum of two to a maximum of 127, with an average of 28 and a median of 19. The word count varied from 33 to 2314 with an average of 476 and a median of 402.

The test data were divided into two sets based on the median word count: one with shorter documents ranging from 28 to 402 words and the other with longer documents ranging from 403 to 2314 words. The CR performance based on document length is presented in Table 6. The performance of CR for short documents appears to be slightly better than that for long documents.

4 | CONCLUSION

This study addressed the CR-M-SpanBERT model for antecedent recognition in NL texts. The CR-M-SpanBERT model consisted of DR, M-SpanBERT, and coreference modules. The DR module parsed an NL sentence to build a dependency tree and then generated a DR. The M-SpanBERT module performed multiple embeddings using NL and DR embeddings, and then performed deep learning using SpanBERT. The coreference module performed span representation using the output of M-SpanBERT, detected mentions among spans, and recognized antecedents among mentions in the NL text.

The effectiveness of the proposed CR-M-SpanBERT model was evaluated by comparing it with another model that utilized SpanBERT as the language model. The experimental results revealed that CR-M-SpanBERT achieved better outcomes in terms of CR accuracy than the other models, except for the model with a large LM. Additionally, the number of epochs required for an average F1 accuracy >75% when the proposed CR-M-SpanBERT model was used was lower than that required when the SpanBERT model was used.

SpanBERT-based CR has the advantage of a significantly smaller parameter size than that of mT5-based CR. However, additional research is required to further improve CR performance.

TABLE 6 Comparison of CR performance based on document length.

Test data	B^3 (F1)	$CEAF_{\varphi 4}$ (F1)	MUC (F1)	Ave. F1
Short documents	79.7	76.5	85.6	80.6
Long documents	79.6	76.3	85.5	80.5

Abbreviation: CR, coreference resolution.

CONFLICT OF INTEREST STATEMENT

The authors declare that there are no conflicts of interest.

ORCID

Joon-young Jung https://orcid.org/0000-0001-6964-4005

REFERENCES

1. F. M. Suchanek, G. Kasneci, and G. Weikum, *YAGO: a core of semantic knowledge*, (Proc. Int. Conf. WWW, Banff, Canada), 2007, pp. 697–706.
2. D. Vrandečić and M. Krötzsch, *Wikidata: a free collaborative knowledgebase*, Communications of the ACM **57** (2014), no. 10, 78–85.
3. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, *DBpedia: a nucleus for a web of open data*, (Proc. Int. Semantic Web Conf., Busan, Republic of Korea), 2007, pp. 722–735.
4. J. Jung, *DG-based SPO tuple recognition using self-attention M-bi-LSTM*, ETRI J. **44** (2022), no. 3, 438–449.
5. N. Q. Luong and A. Popescu-Belis, *Improving pronoun translation by modeling coreference uncertainty*, (Proc. Conf. Machine Translation, Berlin, Germany), 2016, pp. 12–20.
6. A. Mitra and C. Baral, *Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning*, (Proc. AAAI Conf. on Artificial Intelligence, Phoenix, AZ, USA), 2016, pp. 2779–2785.
7. C. Aone and S. W. Bennett, *Evaluating automated and manual acquisition of anaphora resolution strategies*, (Proc. Association for Computational Linguistics, Cambridge, MA, USA), 1995, pp. 122–129.
8. J. McCarthy and W. Lehnert, *Using decision trees for coreference resolution*, (Proc. Int. Conf. on Artificial Intelligence, Montreal, Canada), 1995, pp. 1050–1055.
9. W. M. Soon, H. T. Ng, and D. C. Y. Lim, *A machine learning approach to coreference resolution of noun phrases*, Comput. Linguist. **27** (2001), no. 4, 521–544.
10. V. Ng and C. Cardie, *Improving machine learning approaches to coreference resolution*, (Proc. Association for Computational Linguistic, Philadelphia, PA, USA), 2002, pp. 104–111.
11. E. Bengtson and D. Roth, *Understanding the value of features for coreference resolution*, (Conf. on Empirical Methods in Natural Language Processing, Honolulu, HI, USA), 2008, pp. 294–303.
12. A. Björkelund and J. Kuhn, *Learning structured perceptrons for coreference resolution with latent antecedents and non-local features*, (Proc. Association for Computational Linguistics, Baltimore, MD, USA), 2014, pp. 47–57.
13. K. Clark and C. D. Manning, *Entity-centric coreference resolution with model stacking*, (Proc. Association for Computational Linguistics and Int. Joint Conf. on Natural Language Processing, Beijing, China), 2015, pp. 1405–1415.
14. S. Wiseman, A. M. Rush, and S. M. Shieber, *Learning global features for coreference resolution*, (Proc. North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA), 2016, pp. 994–1004.
15. G. Durrett and D. Klein, *Easy victories and uphill battles in coreference resolution*, (Proc. Empirical Methods in Natural Language Processing, Seattle, WA, USA), 2013, pp. 1971–1982.
16. S. Wiseman, A. M. Rush, S. Shieber and J. Weston, *Learning anaphoricity and antecedent ranking features for coreference resolution*, (Proc. Association for Computational Linguistics and Int. Joint Conf. on Natural Language Processing, Beijing, China), 2015, pp. 1416–1426.
17. A. Marasovic, L. Born, J. Opitz and A. Frank, *A mention-ranking model for abstract anaphora resolution*, (Proc. Empirical Methods in Natural Language Processing, Copenhagen, Denmark), 2017, pp. 221–232.
18. K. Lee, L. He, and L. Zettlemoyer, *Higher-order coreference resolution with coarse-to-fine inference*, (Proc. North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA), 2018, pp. 687–692.
19. B. Kantor and A. Globerson, *Coreference resolution with entity equalization*. (Proc. Association for Computational Linguistics, Florence, Italy), 2019, pp. 673–677.
20. M. Joshi, O. Levy, L. Zettlemoyer and D. Weld, *BERT for coreference resolution: baselines and analysis*, (Proc. Empirical Methods in Natural Language Processing and Int. Joint Conf. on Natural Language Processing, Hong Kong, China), 2019, pp. 5803–5808.
21. M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, *SpanBERT: improving pre-training by representing and predicting spans*, Trans. Assoc. Comput. Linguist. **8** (2020), 64–77.
22. C. Park, J. Lim, J. Ryu, H. Kim, and C. Lee, *Simple and effective neural coreference resolution for Korean language*, ETRI J. **43** (2021), no. 6, 1038–1048.
23. Y. Kirstain, O. Ram and O. Levy, *Coreference resolution without span representations*, (Proc. Association for Computational Linguistics and Int. Joint Conf. on Natural Language Processing, Bangkok, Thailand), 2021, pp. 14–19.
24. I. Beltagy, M. E. Peters and A. Cohan, *Longformer: the long-document transformer*, arXiv , 2020. https://doi.org/10.48550/arXiv.2004.05150
25. T. Liu, Y. E. Jiang, N. Monath, R. Cotterell, M. Sachan *Autoregressive structured prediction with language models*, (Findings of the Association for Computational Linguistics: EMNLP, Abu Dhabi, United Arab Emirates), 2022, pp. 993–1005.
26. B. Bohnet, C. Alberti, and M. Collins, *Coreference resolution through a seq2seq transition-based system*, Trans. Assoc. Comput. Linguist **11** (2023), 212–226.
27. J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, *BERT: pre-training of deep bidirectional transformers for language understanding*, (Proc. North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA), 2019, pp. 4171–4186.
28. T. Mikolov, K. Chen, G. S. Corrado and J. Dean, *Efficient estimation of word representations in vector space*, arXiv, 2013, https://doi.org/48550/arXiv.1301.3781
29. T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, *Distributed representations of words and phrases and their compositionality*, (Proc. Neural Information Processing Systems, Stateline, NV, USA), 2013, pp. 3111–3119.
30. J. Pennington, R. Socher, and C. D. Manning, *GloVe: global vectors for word representation*, (Proc. Empirical Methods in Natural Language Processing, Doha, Qatar), 2014, pp. 1532–1543.

31. D. Chen and C. D. Manning, *A fast and accurate dependency parser using neural networks*, (Proc. Empirical Methods in Natural Language Processing, Doha, Qatar), 2014, pp. 740–750.
32. J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, D. Zeman, *Universal dependencies v1: a multilingual treebank collection*, (Proc. Int. Conf. on Language Resources and Evaluation, Portorož, Slovenia), 2016, pp. 1659–1666.
33. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N Gomez, L. Kaiser, and I. Polosukhin, *Attention is all you need*, (Proc. Neural Information Processing Systems, Long Beach, CA, USA), 2017, pp. 6000–6010.
34. N. S. Moosavi and M. Strube, *Which coreference evaluation metric do you trust? A proposal for a link-based entity aware metric*, (Proc. Association for Computational Linguistics, Berlin, Germany), 2016, pp. 632–642.
35. L. Miculicich and J. Henderson, *Graph refinement for coreference resolution*, (Proc. Association for Computational Linguistics, Dublin, Ireland), 2022, pp. 2732–2742.

AUTHOR BIOGRAPHY

Joon-young Jung received his BS and MS degrees in Computer Network Engineering from Soongsil University, Seoul, Republic of Korea, in 1996 and 2000, respectively, and his PhD degree in Information Communications Engineering from Chungnam National University, Daejeon, Republic of Korea, in 2015. Since 2000, he has been a research engineer at the Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea. His main research interests are natural language processing and visual intelligence.

How to cite this article: J. Jung, *CR-M-SpanBERT: Multiple embedding-based DNN coreference resolution using self-attention SpanBERT*, ETRI Journal **46** (2024), 35–47, DOI 10.4218/etrij.2023-0308