symmetry

MDPI

Article

Semantic Relation Classification via Bidirectional LSTM Networks with Entity-Aware Attention Using Latent Entity Typing

Joohong Lee [1,2], Sangwoo Seo [1] and Yong Suk Choi [1,*]

[1] Department of Computer Science and Engineering, Hanyang University, Seoul 04763, Korea; roomylee@hanyang.ac.kr (J.L.); ssw1591@hanyang.ac.kr (S.S.)
[2] Pingpong AI Research, Scatter Lab, Seoul 06103, Korea
* Correspondence: cys@hanyang.ac.kr; Tel.: +82-2-2220-1139

Received: 23 April 2019; Accepted: 5 June 2019; Published: 13 June 2019

Abstract: Classifying semantic relations between entity pairs in sentences is an important task in natural language processing (NLP). Most previous models applied to relation classification rely on high-level lexical and syntactic features obtained by NLP tools such as WordNet, the dependency parser, part-of-speech (POS) tagger, and named entity recognizers (NER). In addition, state-of-the-art neural models based on attention mechanisms do not fully utilize information related to the entity, which may be the most crucial feature for relation classification. To address these issues, we propose a novel end-to-end recurrent neural model that incorporates an entity-aware attention mechanism with a latent entity typing (LET) method. Our model not only effectively utilizes entities and their latent types as features, but also builds word representations by applying self-attention based on symmetrical similarity of a sentence itself. Moreover, the model is interpretable by visualizing applied attention mechanisms. Experimental results obtained with the SemEval-2010 Task 8 dataset, which is one of the most popular relation classification tasks, demonstrate that our model outperforms existing state-of-the-art models without any high-level features.

Keywords: relation extraction; entity-aware attention; latent entity typing; end-to-end learning; visualization

1. Introduction

Classifying semantic relations between entity pairs in sentences plays a crucial role in various NLP tasks, such as information extraction, question answering, and knowledge base population [1]. A task in relation classification is defined as predicting a semantic relationship between two tagged entities in a sentence. For example, given a sentence with the tagged entity pair, *crash* and *attack*, this sentence is classified into the relation *Cause-Effect(e1,e2)*, one of the pre-defined relation classes in the SemEval-2010 Task 8 [2], as shown as Figure 1.

Sentence :		
"The train <e1>*crash*</e1> was caused by terrorist <e2>*attack*</e2>"		
Entity 1 : *crash*	**Entity 2** : *attack*	**Relation** : *Cause-Effect(e1,e2)*

Figure 1. A Sample of Relation Classification.

The classification models relying on high-level lexical and syntactic features obtain by NLP tools suffer from the propagation of implicit error of the tools and they are computationally expensive. Therefore, many recent studies have proposed end-to-end neural models without high-level features.

Among these, attention-based neural models that focus on semantically important parts of a sentence show state-of-the-art results in a lot of NLP tasks. Given that these models are mainly proposed for solving translation and language modeling tasks, they are not capable of fully utilizing the information of tagged entities in the relation classification task. However, tagged entity pairs could provide powerful hints for solving relation classification task. For example, even if we do not consider other words except *crash* and *attack*, we intuitively know that the entity pair has a relation *Cause-Effect(e1,e2)* instead of *Component-Whole(e1,e2)* from Figure 1.

To address these issues, we propose a novel end-to-end recurrent neural model, which incorporates an entity-aware attention mechanism with a latent entity typing (LET). To capture the context of sentences, we obtained word representations by self-attention mechanisms and build the recurrent neural architecture with bidirectional long short-term memory (LSTM) networks. Entity-aware attention focuses on the most important semantic information by considering entity pairs with along their latent type representation obtained by LET which constructs a latent type vector of entity according to its characteristic.

The contributions of our work are summarized as follows: (1) We propose a novel end-to-end recurrent neural model and an entity-aware attention mechanism with a LET which focuses on semantic information of entities and their latent types. (2) Our model obtains an 85.2% F1-score using the SemEval-2010 Task 8 dataset and outperforms existing state-of-the-art models without any high-level features. (3) We show that our model is more interpretable by visualizing the attention mechanisms applied to our model.

2. Related Work

Several studies have been conducted to solve the relation classification task. Early methods used handcrafted features through a series of NLP tools or manually designed kernels [3]. These approaches use high-level lexical and syntactic features obtained from NLP tools and manually designed kernels, but the classification models relying on such features suffer from propagation of the implicit error contained the tools [4].

On the other hand, deep neural networks have outperformed previous models by using handcrafted features. In particular, many researchers attempted to solve this problem based on end-to-end models using only raw sentences and pre-trained word representations learned by Skip-gram and Continuous Bag-of-Words [5–7]. Zeng et al. employed a deep convolutional neural network (CNN) for extracting lexical and sentence level features [8]. Dos Santos et al. proposed a model for learning vector for each relation class using ranking loss to reduce the impact of artificial classes [9]. Zhang and Wang used the bidirectional recurrent neural network (RNN) to learn long-term dependency between entity pairs [10]. Furthermore, Zhang et al. proposed a bidirectional LSTM network (BLSTM) utilizing the position of words, POS tags, named entity information, and dependency parse [11]. This model-resolved vanishing gradient problem appeared in RNNs by using a BLSTM.

Recently, some researchers have proposed attention-based models that can focus to the most important semantic information in a sentence. Zhou et al. combined attention mechanisms with a BLSTM [4]. Xiao and Liu split the sentence into two entities and used two attention-based BLSTMs hierarchically [12]. Shen and Huang proposed an attention-based CNN using word-level attention mechanism that can focus on more influential parts in the sentence [13].

In contrast with the end-to-end models, several studies proposed models utilizing the shortest dependency path (SDP) between entity pairs of dependency parse trees. The SDP-LSTM model proposed by Yan et al. and the deep recurrent neural networks (DRNNs) model proposed by Xu et al. eliminate irrelevant words out of SDP and use the neural network based on meaningful words composing the SDP [14,15].

3. Model

In this section, we introduce in detail a novel recurrent neural model that incorporates an entity-aware attention mechanism with a LET method. As shown in Figure 2, our model consists of

four main components: (1) Word Representation,mapping each word in a sentence into corresponding vector representations. (2) Self-Attention, capturing the meaning of the correlation between words based on multi-head attention [16]. (3) BLSTM, which sequentially encodes the representations of the self-attention layer. (4) Entity-aware Attention, which calculates attention weights with respect to the entity pairs, word positions relative to these pairs, and their type representations obtained by our LET that constructs a latent type representation vector of entity according to the characteristic of each entity based on attention mechanism. The obtained features are then averaged along time steps to produce sentence-level features.

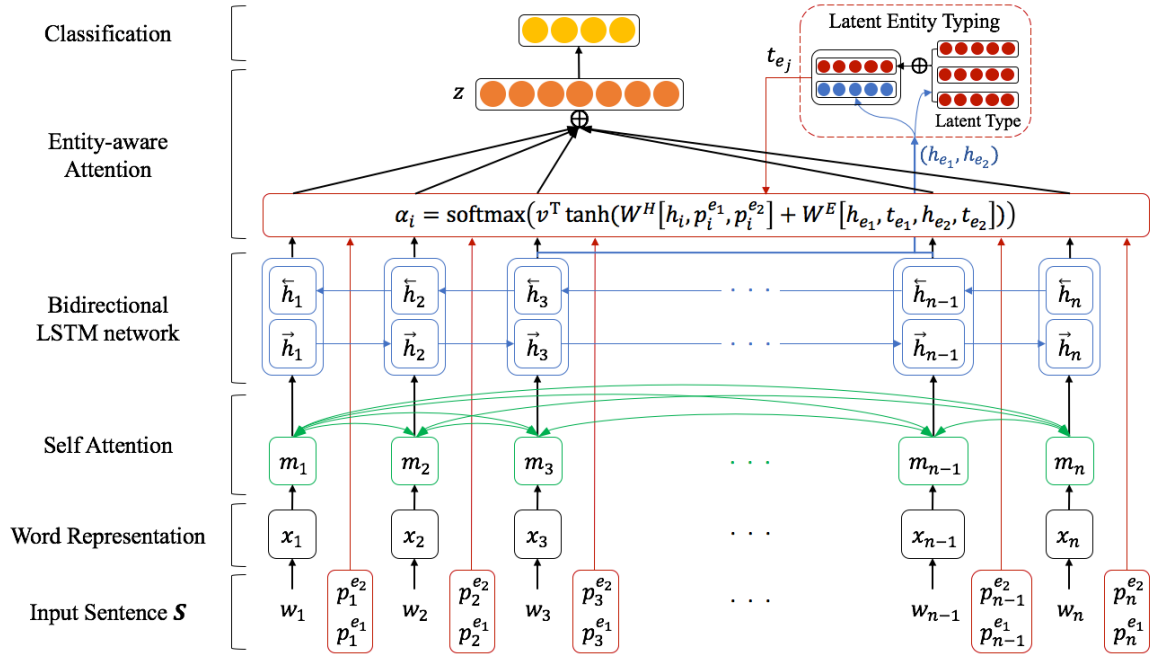

Figure 2. The architecture of our model (best viewed in color). Entity 1 and 2 corresponds to the 3 and $(n-1)$-th words, respectively, which are fed into the LET.

3.1. Word Representations

Let an input sentence be denoted by $S = \{w_1, w_2, ..., w_n\}$, where n denotes the number of words. We transform each word into a vector representation by looking up the word embedding matrix $W_{word} \in \mathbb{R}^{d_w \times |V|}$, where d_w is the dimension of the vector and $|V|$ is the size of vocabulary. Then, the word representations $X = \{x_1, x_2, ..., x_n\}$ are obtained by mapping w_i, the i-th word, to a column vector $x_i \in \mathbb{R}^{d_w}$, which are then fed into the next layer.

3.2. Self-Attention

The word representations are fixed for each word, even though meanings of words vary depending on the context. Many neural models encoding a sequence of words may expect to learn implicitly of the contextual meaning, but they may not learn well because of long-term dependency problems [17]. In order for the representation vectors to capture the meaning of the words with consideration of the context, we employ a self-attention, a special case of attention mechanisms, which only requires a single sequence and builds a symmetrical similarity matrix of a sentence itself. Self-attention has been widely applied to many NLP tasks such as machine translation, language understanding, and semantic role labeling [16,18,19].

We adopt the multi-head attention formulation [16], one of the methods for implementing the self-attention. Figure 3 illustrates the multi-head attention mechanism that consists of several linear transformations and the scaled dot-product attention mechanism corresponding to the center block of the figure.

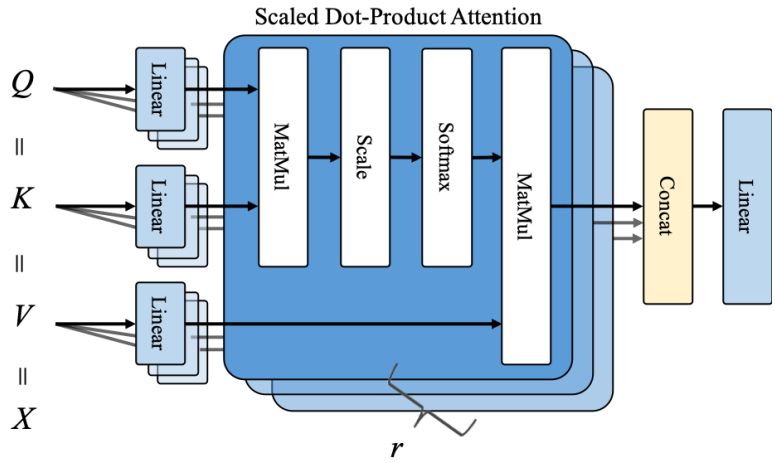

Figure 3. Multi-Head Self-Attention. For self-attention, the Q (query), K (key), and V (value), inputs of multi-head attention, should be the same vectors. In our work, they are equivalent to X, the word representation vectors.

Given a matrix of n vectors, query Q, key K, and value V, the scaled dot-product attention is calculated using the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_w}})V \tag{1}$$

In multi-head attention, the scaled dot-product attention with linear transformations is performed on r parallel heads to pay attention to different parts. Then, the formulation of multi-head attention is defined by the follows:

$$\text{MultiHead}(Q, K, V) = W^M[head_1; ...; head_r] \tag{2}$$

$$head_i = \text{Attention}(W_i^Q Q, W_i^K K, W_i^V V) \tag{3}$$

where [;] depicts row concatenation and r is the number of heads. The weights $W^M \in \mathbb{R}^{d_w \times d_w}$, $W_i^Q \in \mathbb{R}^{d_w/r \times d_w}$, $W_i^K \in \mathbb{R}^{d_w/r \times d_w}$, and $W_i^V \in \mathbb{R}^{d_w/r \times d_w}$ are learnable parameters for linear transformation. W^M denotes concatenation outputs of scaled dot-product attention, and the other three above-mentioned weights denote query, key, and value of i-th head, respectively.

Because our work requires self-attention, the input matrices of multi-head attention, Q, K, and V are all equivalent to X, the word representation vectors. As a result, outputs of multi-head attention are denoted by $M = \{m_1, m_2, ..., m_n\} = \text{MultiHead}(X, X, X)$, where m_i is the output vector corresponding to i-th word. The output of the self-attention layer is the sequence of representations that include informative factors of the input sentence.

3.3. Bidirectional LSTM Network

To sequentially encode the output of the self-attention layer, we use a BLSTM [20,21] consisting of two sub LSTM networks: a forward LSTM network, which encodes the context of an input sentence, and a backward LSTM network, which encodes the context of the reversed sentence. More formally, BLSTM works as follows:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(m_t) \tag{4}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(m_t) \tag{5}$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{6}$$

The representation vectors M obtained from the self-attention layer are forwarded into to the network step by step. At the time step t, the hidden state $h_t \in \mathbb{R}^{2d_h}$ of a BLSTM is obtained by concatenating $\overrightarrow{h_t} \in \mathbb{R}^{d_h}$, the hidden state of forward LSTM network, and $\overleftarrow{h_t} \in \mathbb{R}^{d_h}$, the hidden state of the backward LSTM network, where d_h is dimension of each LSTM's state.

3.4. Entity-Aware Attention Mechanism

Many models with attention mechanisms achieved state-of-the-art performance in various NLP tasks. In relation classification, these models also show good performance, but they do not fully utilize entity information which could be powerful hints for solving the task. To address this issue, we propose a novel entity-aware attention mechanism for fully utilizing informative factors in given entity pairs. Entity-aware attention uses two additional features: (1) Relative position features; (2) Entity features with latent types. The relative position features help to build contextual representation that takes into account entity pairs [22]. On the other hand, the entity features are composed of BLSTM's hidden states corresponding to positions of entity pairs, and their latent type vectors obtained by our latent entity typing (LET) method. These are formulated as shown in left and right term of Equation (7), respectively. The final sentence representation z, resulting from this attention mechanism, is computed as follows:

$$u_i = \tanh(W^H[h_i; p_i^{e_1}; p_i^{e_2}] + W^E[h_{e_1}; t_1; h_{e_2}; t_2]) \tag{7}$$

$$\alpha_i = \frac{\exp(v^\top u_i)}{\sum_{j=1}^{n} \exp(v^\top u_j)} \tag{8}$$

$$z = \sum_{i=1}^{n} \alpha_i h_i \tag{9}$$

3.4.1. Relative Position Features

In relation classification, the position of each word relative to the entities has been widely used for word representations [1,8,13]. Recently, position-aware attention was published as a way to use relative position features more effectively [22]. It is a variant of the attention mechanisms, which use not only outputs from BLSTM but also the relative position features when calculating attention weights.

We adopt this method with a slightly modification as shown in Equation (7). In the equation, $p_i^{e_1} \in \mathbb{R}^{d_p}$ and $p_i^{e_2} \in \mathbb{R}^{d_p}$ corresponds to the position of the i-th word relative to the first entity (e_1-th word) and second entity (e_2-th word) in a sentence respectively, where $e_{j\in\{1,2\}}$ is an index of the j-th entity. Similar to word embeddings, relative positions are converted to vector representations by looking up in the learnable embedding matrix $W_{pos} \in \mathbb{R}^{d_p \times (2L-1)}$, where d_p is the dimension of the relative position vectors and L is the maximum sentence length. Finally, representations of the BLSTM layer consider the context and the positional relationship of entities by concatenating h_i, $p_i^{e_1}$, and $p_i^{e_2}$. The representation is linearly transformed by $W^H \in \mathbb{R}^{d_a \times (2d_h + 2d_p)}$, as in the Equation (7).

3.4.2. Entity Features with Latent Types

Since entity pairs present powerful hints for solving the relation classification task, we involve representation of entity pairs and their types in our attention mechanism more directly. The representation of entity is obtained from BLSTM's hidden state corresponding to its position. These are denoted by $h_{e_j} \in \mathbb{R}^{2d_h}$, where e_j is the index of j-th entity. To leverage richer entity information, many models for various NLP tasks [23] use predefined entity types such as *PER*, *LOC*, and *ORG* obtained by NER [24]. However, it is not appropriate for our approach based on end-to-end learning to use it directly. To address this issue, we propose a latent entity typing(LET) that trains latent representations of entity types instead of using predefined entity types from NER. LET builds the latent variable $c \in \mathbb{R}^{2d_h \times K}$ that represents unrevealed entity types on the latent vector space, where K is the number of latent entity

types. The latent type representation t_j of j-th entity is calculated by weighting c against the similarity to h_{e_j}, representation of j-th entity, based on an attention mechanism. In Figure 2, the box at the top right illustrates mechanism of LET, and the mathematical formulation is as follows:

$$a_i^j = \frac{\exp((h_{e_j})^\top c_i)}{\sum_{k=1}^{K} \exp((h_{e_j})^\top c_k)} \tag{10}$$

$$t_{j\in\{1,2\}} = \sum_{i=1}^{K} a_i^j c_i \tag{11}$$

where c_i is the i-th latent type vector. Our LET is inspired by latent topic clustering, a method for predicting the latent topic of texts in the question answering task [25].

Consequently, entity features are constructed by concatenating the hidden states corresponding to entity positions and latent type representations of entity pairs. After linear transformation of the entity features, they are added to the representations of the BLSTM layer with relative position feature shown as in Equation (7), and the representation of sentence $z \in \mathbb{R}^{2d_h}$ is computed by Equations from (7) to (9).

3.5. Classification and Training

The sentence representation obtained from the entity-aware attention z is fed into a fully connected softmax layer for classification. It generates the conditional probability $p(y|S,\theta)$ over all relation types:

$$p(y|S,\theta) = \text{softmax}(W^O z + b^O) \tag{12}$$

where y is a target relation class and S is the input sentence. The θ denotes learnable parameters in the entire network including $W^O \in \mathbb{R}^{|R|\times 2d_h}$, $b^O \in \mathbb{R}^{|R|}$, where $|R|$ is the number of relation classes. A loss function $\mathcal{L}$ is the cross entropy between the predictions and the ground truths, which is defined as:

$$\mathcal{L} = -\sum_{i=1}^{|D|} \log p(y^{(i)}|S^{(i)},\theta) + \lambda||\theta||_2^2 \tag{13}$$

where $|D|$ is the size of training dataset and $(S^{(i)}, y^{(i)})$ is the i-th sample in the dataset. We minimize the loss $\mathcal{L}$ using the AdaDelta optimizer [26] to compute the parameters θ of our model.

To alleviate overfitting, we constrain the L2 regularization with the coefficient λ [27]. In addition, the dropout method is applied after the word embedding, LSTM network, and entity-aware attention to prevent co-adaptation of feature detectors [28,29].

4. Experiments

4.1. Dataset and Evaluation Metrics

We evaluate our model on the SemEval-2010 Task 8 dataset, a commonly used benchmark for relation classification [2], and compare the results with the state-of-the-art models in this area. The dataset contains 10 distinguished relations, *Cause-Effect*, *Instrument-Agency*, *Product-Producer*, *Content-Container*, *Entity-Origin*, *Entity-Destination*, *Component-Whole*, *Member-Collection*, *Message-Topic*, and *Other*. The former 9 relations have two directions, whereas *Other* is not directional, such the total number of relations is 19. There are 10,717 annotated sentences which consist of 8000 samples for training and 2717 samples for testing. We adopt the official evaluation metric of the SemEval-2010 Task 8 dataset, which is based on the macro-averaged F1-score (excluding *Other*), and takes into account the directionality.

4.2. Implementation Details

We tune the hyperparameters of our model on the development set of randomly sampled 800 sentences for validation. The best hyperparameters in our proposed model are listed in Table 1.

Table 1. Hyperparameters of Our Model.

Hyper-parameter	Description	Value
d_w	Size of Word Embeddings	300
r	Number of Heads	4
d_h	Size of Hidden Layer	300
d_p	Size of Position Embeddings	50
d_a	Size of Attention Layer	50
K	Number of Latent Entity Types	3
batch_size	Size of Mini-Batch	20
η	Initial Learning Rate	1.0
dropout rate	Word Embedding layer	0.3
	BLSTM layer	0.3
	Entity-aware Attention layer	0.5
λ	L2 Regularization Coefficient	10^{-5}

We use pre-trained weights of the publicly available GloVe model [7] to initialize word embeddings in our model, and other weights, which are randomly initialized from the zero-mean Gaussian distribution [30].

4.3. Experimental Results

Table 2 compares our Entity-aware Attention LSTM model with state-of-the-art models on this relation classification dataset. We divide the models into three groups; *Non-Neural Model*, *SDP-based Model*, and *End-to-End Model*.

Table 2. Comparison with Previous Results on SemEval-2010 Task 8 test dataset, where the WN, WAN, PF, and DEP are WordNet (hypernyms), words around nominals, position features, and dependency features, respectively.

	Model	F_1-Score
Non-Neural Model	SVM [3]	82.2
SDP-based Model	MVRNN [31]	82.4
	FCM [32]	83.0
	DepNN [33]	83.6
	depLCNN + NS [34]	**85.6**
	SDP-LSTM [14]	83.7
	DRNNs [15]	**86.1**
End-to-End Model	CNN [8]	82.7
	CR-CNN [9]	84.1
	Attention-CNN [13]	84.3
	+ POS, WN, WAN	**85.9**
	BLSTM [11]	82.7
	+ PF, POS, NER, DEP, WN	84.3
	Attention-BLSTM [4]	84.0
	Hier-BLSTM [35]	84.3
	Our Model	**84.7**
	+ Latent Entity Typing	**85.2**

First, the SVM [3], *Non-Neural Model*, was on top of the SemEval-2010 task during the official competition period. To train an SVM classifier, it used many handcrafted features such as WordNet, ProBank, and FrameNet that are usually not robust and computationally intensive [1]. As a result, it achieved an F1-score of 82.2%. The second is a *SDP-based Model* such as the MVRNN [31], FCM [32], DepNN [33], depLCNN + NS [34], SDP-LSTM [14], and DRNNs [15]. The SDP is a reasonable feature for detecting the semantic structure of sentences. Actually, SDP-based models show high performance, but SDP may not always be accurate and the parsing time exponentially increases with long sentences [36]. The last model is the *End-to-End Model*, where automatically learned internal

representations can occur between original inputs and final outputs in deep learning. For this task, there are CNN-based models such as CNN [1,8], CR-CNN [9], and Attention-CNN [13] and RNN-based models such as BLSTM [11], Attention-BLSTM [4], and Hierarchical-BLSTM (Hier-BLSTM) [35].

Our proposed model with LET achieves a F_1-score of 85.2% which outperforms all competing state-of-the-art approaches except depLCNN + NS, DRNNs, and Attention-CNN. However, these models have a defect that is heavy dependence on high-level lexical features such as WordNet, dependency parse trees, POS tags, and NER tags from NLP tools.

Experimental results in the Table 2 show that the LET is effective for relation classification by improving the performance by 0.5% (absolutely) on the test dataset. To apply the LET, the number of latent types, one of the model hyperparameters, should be fixed, so we have conducted an additional experiment to obtain the optimal number of them, as shown in Figure 4.

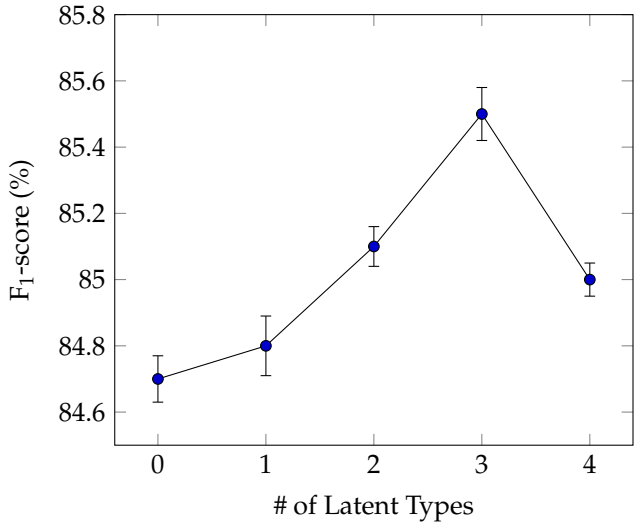

Figure 4. Performance by the Number of Latent Types.

5. Analysis and Discussion

5.1. Ablation Study

We perform an ablation study on development set of SemEval-2010 Task 8 in order to evaluate the impact of each feature and component applied to our entity-aware attention model, as shown in Table 3. We set the attention-based bidirectional LSTM [4] as the baseline model, and it achieves a F_1-score of 82.8%. The F_1-score improves remarkably when new features and components are added. We use GloVe model, one of the most popular pre-trained word embeddings, to initialize the embeddings, and it contributes about F_1-score of 1.2%. The newly added components, self-attention, entity-aware attention, and latent entity typing, in our proposed model achieved a F_1-score of 85.5% which is 1.3% higher than baseline model with pre-trained embeddings and relative position features. These results show that our proposals are effective enough.

Table 3. Ablation study for the contribution of each feature applied to our model.

Model	**Dev F_1**
Baseline(Att-BLSTM)	82.8
+ Pre-trained word embeddings	84.0
+ Relative position features	84.2
+ Self-attention	84.5
+ Entity-aware attention	84.8
+ Latent entity typing	85.5

5.2. Visualization

There are two different visualizations that demonstrate that the model proposed in this paper is interpretable. First, the visualization of self-attention shows the focus of each word as part of the sentence. By showing the words that the entity pair deals with, we can find the words that well represent the relation between them. Second, the entity-aware attention visualization shows where the model pays attention to a sentence. This visualization result highlights important words in a sentence, which are usually important keywords for classification.

5.2.1. Self-Attention

We can obtain richer word representations by using self-attention. These word representations are considered as the context, based on the correlation between words in a sentence. The Figure 5 illustrates the results of self-attention in the sentence, "*the <e1> pollution </e1> was caused by the <e2> shipwrek </e2>*", which is labeled as *Cause-Effect(e2,e1)*. There are visualizations of the two heads in the multi-head attention applied for self-attention. The color density indicates attention values, results of Equation 1, which implies how much an entity focuses on each word in a sentence.

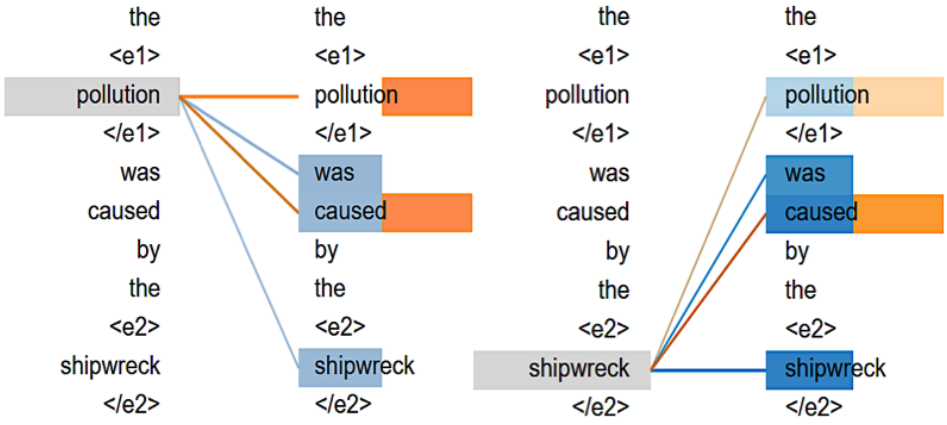

Figure 5. Visualization of Self-Attention.

In Figure 5, the left side represents the words that *pollution*, the first entity, focuses on and the right represents the words that *shipwreck*, the second entity, focuses on. We can recognize that the entity pair is commonly concentrated on *was*, *caused*, and each other. Actually, these words play the most important role in semantically predicting the *Cause-Effect(e2,e1)*, which is the relation class of this entity pair.

5.2.2. Entity-Aware Attention

Figure 6 shows where the model focuses on the sentence to compute relations between entity pairs, which is the result of visualizing alpha vectors in Equation (8). The important words in a sentence are highlighted in yellow, meaning the lighter the color, the more important it is to the sentence. For example, in the first sentence, the *inside* is strongly highlighted, which presents the best word representing the relation *Component-whole(e1,e2)* between the given entity pair. As another example, in the third sentence, the highlighted words, *assess* and *using*, represent well the relation *Instrument-Agency(e2,e1)* between entity pair, *analysts* and *frequency*. We can see that the word *using* is more highlighted than the word *assess*, because the former represents the relation better.

Sentence	Entity 1	Entity 2	Relation
the <e1> castle </e1> was inside a <e2> museum </e2>	*castle*	*museum*	*Component-Whole(e1,e2)*
the <e1> design </e1> is by my <e2> wife </e2> bianca	*design*	*wife*	*Product-Producer(e1,e2)*
<e1> analysts </e1> assess distribution and changes in distribution over time by using <e2> frequency </e2>	*analysts*	*frequeny*	*Instrument-Agency(e2,e1)*
the prosecution seeks to enter <e1> motives </e1> into gibbs <e2> trial </e2>	*motives*	*trial*	*Entity-Destination(e1,e2)*

Figure 6. Visualization of Entity-aware Attention.

5.3. Assessment of Latent Entity Type

For qualitative evaluation of methods, we extract the 10 nearest entities to each latent type vectors, and Table 4 summarizes the results of the extraction. This allows us to roughly understand what characteristics latent types have. We set the number of latent types to three, because it shows the best performance. The words nearest to the first type vector are related to human's jobs and foods, while the nearest words to the second type vector are related to machines and engineering. On the other hand, in case of type 3, there are many negative meaning words associated with disasters and drugs. As a result, we can see that words of similar type are well clustered around each latent type vector, therefore, the latent type representation of each word obtained by our LET seems to be sufficiently effective to indicate its characteristics.

Table 4. Sets of Entities grouped by Latent Types

Latent Type	10 Nearest Entities to Latent Types
Type 1	*worker, chairman, author, king, potter, cuisine, spaghetti, restaurant, bananas, salas*
Type 2	*systems, engine, trucks, valve, hinge, assembly, woofer, mainspring, circuit, motor*
Type 3	*virus, tsunami, accident, drugs, riot, pandemic, pollution, earthquake, contamination, marijuana*

6. Conclusions

In this paper, we proposed an entity-aware attention mechanism with latent entity typing and a novel end-to-end recurrent neural model, which incorporates this mechanism for relation classification. Our model achieves an 85.2% F1-score in SemEval-2010 Task 8 using only raw sentence and word embeddings without any high-level features from NLP tools, and it outperforms existing state-of-the-art methods. In addition, we show that our model is interpretable by visualizing where the model focuses to solve the relation extraction task. We expect our model to be extended to not only the relation classification task but also to other tasks in which entity plays an important role. In particular, latent entity typing can be effectively applied to sequence modeling task using entity information without NER. In future studies, we plan to propose a new method in question answering or knowledge base population, based on relations between entities extracted from the presented model.

Author Contributions: Conceptualization, J.L.; Methodology, J.L. and S.S.; Software, J.L. and S.S.; Validation, J.L. and S.S.; Formal Analysis, J.L. and S.S.; Investigation, J.L. and S.S.; Resources, J.L. and S.S.; Data Curation, J.L. and S.S.; Writing—Original Draft Preparation, J.L. and S.S.; Writing—Review and Editing, Y.S.C.; Visualization, S.S.; Supervision, Y.S.C.; Project Administration, J.L.; Funding Acquisition, Y.S.C.

Funding: This work was supported by the Technology Innovation Program(No.10060086,10077553) funded by the MOTIE(Ministry of Trade, industry & Energy, Korea), by the MSIT(Ministry of Science and ICT, Korea) under the ITRC(Information Technology Research Center) support program(IITP-2017-0-01642) supervised by the IITP(Institute for Information & communications Technology Promotion), and by the NRF(National Research Foundation) grant funded by the Korea government(MSIT) (No.2018R1A5A7059549).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nguyen, T.H.; Grishman, R. Relation extraction: Perspective from convolutional neural networks. In Proceedings of the NAACL Workshop on Vector Space Modeling for Natural Language Processing, Denver, CO, USA, 5 June 2015; pp. 39–48.
2. Hendrickx, I.; Kim, S.N.; Kozareva, Z.; Nakov, P.; Ó Séaghdha, D.; Padó, S.; Pennacchiotti, M.; Romano, L.; Szpakowicz, S. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions, Linguistics, Boulder, CO, USA, 4 June 2009; pp. 94–99.
3. Rink, B.; Harabagiu, S. Utd: Classifying semantic relations by combining lexical and semantic resources. In Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, 15–16 July 2010; pp. 256–259.
4. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 207–212.
5. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 3111–3119.
6. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
7. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
8. Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. Relation classification via convolutional deep neural network. In Proceedings of the COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 2335–2344.
9. Santos, C.N.D.; Xiang, B.; Zhou, B. Classifying Relations by Ranking with Convolutional Neural Networks. *arXiv* **2015**, arXiv:1504.06580.
10. Zhang, D.; Wang, D. Relation classification via recurrent neural network. *arXiv* **2015**, arXiv:1508.01006.
11. Zhang, S.; Zheng, D.; Hu, X.; Yang, M. Bidirectional long short-term memory networks for relation classification. In Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation, Shanghai, China, 30 October–1 November 2015; pp. 73–78.
12. Xiao, M.; Liu, C. Semantic relation classification via hierarchical recurrent neural network with attention. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 1254–1263.
13. Huang, X. Attention-based convolutional neural network for semantic relation extraction. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 2526–2536.
14. Xu, Y.; Mou, L.; Li, G.; Chen, Y.; Peng, H.; Jin, Z. Classifying relations via long short term memory networks along shortest dependency paths. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1785–1794.
15. Xu, Y.; Jia, R.; Mou, L.; Li, G.; Chen, Y.; Lu, Y.; Jin, Z. Improved relation classification by deep recurrent neural networks with data augmentation. *arXiv* **2016**, arXiv:1601.03651.
16. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
17. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef] [PubMed]
18. Shen, T.; Zhou, T.; Long, G.; Jiang, J.; Pan, S.; Zhang, C. Disan: Directional self-attention network for rnn/cnn-free language understanding. *arXiv* **2017**, arXiv:1709.04696.
19. Tan, Z.; Wang, M.; Xie, J.; Chen, Y.; Shi, X. Deep semantic role labeling with self-attention. *arXiv* **2017**, arXiv:1712.01586.

20. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef] [PubMed]
21. Graves, A.; Mohamed, A.R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.
22. Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; Manning, C.D. Position-aware attention and supervised data improve slot filling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 35–45.
23. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
24. Sang, E.F.; De Meulder, F. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv* **2003**, arXiv:cs/0306050.
25. Yoon, S.; Shin, J.; Jung, K. Learning to Rank Question-Answer Pairs Using Hierarchical Recurrent Encoder with Latent Topic Clustering. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 1575–1584.
26. Zeiler, M.D. ADADELTA: An adaptive learning rate method. *arXiv* **2012**, arXiv:1212.5701.
27. Ng, A.Y. Feature selection, L 1 vs. L 2 regularization, and rotational invariance. In Proceedings of the Twenty-First International Conference on Machine Learning, Banff, Alberta, Canada, 4–8 July 2004; p. 78.
28. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv* **2012**, arXiv:1207.0580.
29. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent neural network regularization. *arXiv* **2014**, arXiv:1409.2329.
30. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
31. Socher, R.; Huval, B.; Manning, C.D.; Ng, A.Y. Semantic compositionality through recursive matrix-vector spaces. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Korea, 12–14 July 2012; pp. 1201–1211.
32. Yu, M.; Gormley, M.; Dredze, M. Factor-based compositional embedding models. In Proceedings of the NIPS Workshop on Learning Semantics, Montreal, QC, Canada, 12 December 2014; pp. 95–101.
33. Liu, Y.; Wei, F.; Li, S.; Ji, H.; Zhou, M.; Wang, H. A dependency-based neural network for relation classification. *arXiv* **2015**, arXiv:1507.04646.
34. Xu, K.; Feng, Y.; Huang, S.; Zhao, D. Semantic relation classification via convolutional neural networks with simple negative sampling. *arXiv* **2015**, arXiv:1506.07650.
35. Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; Hovy, E. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 1480–1489.
36. Chen, D.; Manning, C. A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October; pp. 740–750.