Semantic Relation Classification via Bidirectional LSTM Networks with Entity-aware Attention using Latent Entity Typing

Joohong Lee* Sangwoo Seo* Yong Suk Choi*†

Abstract

Classifying semantic relations between entity pairs in sentences is an important task in Natural Language Processing (NLP). Most previous models for relation classification rely on the high-level lexical and syntatic features obtained by NLP tools such as WordNet, 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 of entity that may be the most crucial features for relation classification. 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) method. Our model not only utilizes entities and their latent types as features effectively but also is more interpretable by visualizing attention mechanisms applied to our model and results of LET. Experimental results on the SemEval-2010 Task 8, one of the most popular relation classification task, demonstrate that our model outperforms existing state-of-the-art models without any high-level features.

1 Introduction

Classifying semantic relations between entity pairs in sentences plays a vital role in various NLP tasks, such as information extraction, question answering and knowledge base population [14]. A task of relation classification is defined as predicting a semantic relationship between two tagged entities in a sentence.

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.

For example, given a sentence with tagged entity pair, *crash* and *attack*, this sentence is classified into the relation *Cause-Effect(e1,e2)*[1] between the entity pair like Figure 1. A first entity is surrounded by $\langle e1 \rangle$ and $\langle /e1 \rangle$, and a second entity is surrounded by $\langle e2 \rangle$ and $\langle /e2 \rangle$.

Most previous relation classification models rely heavily on high-level lexical and syntactic features obtained from NLP tools such as WordNet, dependency parser, part-of-speech (POS) tagger, and named entity recognizer (NER). The classification models relying on such features suffer from propagation of implicit error of the tools and they are computationally expensive.

Recently, many studies therefore propose end-to-end neural models without the high-level features. Among them, attention-based models, which focus to the most important semantic information in a sentence, show state-of-the-art results in a lot of NLP tasks. Since these models are mainly proposed for solving translation and language modeling tasks, they could not fully utilize the information of tagged entities in relation classification task. However, tagged entity pairs could be powerful hints for solving relation classification task. For example, even if we do not consider other words except the *crash* and *attack*, we intuitively know that the entity pair has a relation *Cause-Effect(e1,e2)*[1] better than *Component-Whole(e1,e2)*[1] in 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 obtain 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 considering entity pairs with word positions relative to these pairs and latent types obtained by LET.

The contributions of our work are summarized as follows: (1) We propose an novel end-to-end recurrent neural model and an entity-aware attention mechanism with a LET which focuses to semantic information of entities and their latent types; (2) Our model obtains 85.2% F1-score in SemEval-2010 Task 8 and it outper-

*Department of Computer Science and Engineering, Hanyang University, Seoul, Republic of Korea, {roomylee, ssw1591, cys}@hanyang.ac.kr

†Corresponding author.

[1]It is one of the pre-defined relation classes in the SemEval-2010 Task 8 [6].

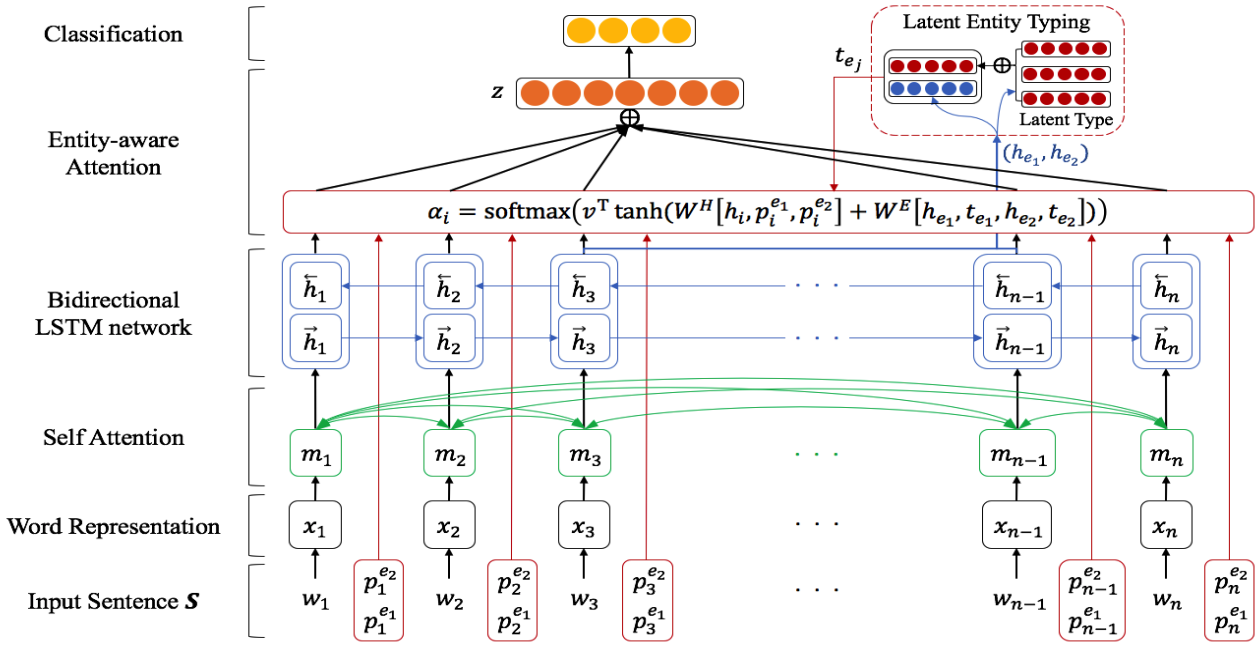

Figure 2: The architecture of our model (best viewed in color). Entity 1 and 2 corresponds to the 3 and $(n-1)$-th words, respectively, which are fed into the LET.

forms existing state-of-the-art models without any high-level features; (3) We show that our model is more interpretable since it's decision making process could be visualized with self attention, entity-aware attention, and LET.

2 Related Work

There are several studies for solving relation classification task. Early methods used handcrafted features through a series of NLP tools or manually designing kernels [16]. These approaches use high-level lexical and syntactic features obtained from NLP tools and manually designing kernels, but the classification models relying on such features suffer from propagation of implicit error of the tools.

On the other hands, deep neural networks have shown outperform previous models using handcraft features. Especially, many researches tried to solve the 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 [12, 11, 15]. Zeng et al. employed a deep convolutional neural network (CNN) for extracting lexical and sentence level features [30]. Dos Santos et al. proposed model for learning vector of each relation class using ranking loss to reduce the impact of artificial classes [2]. Zhang and Wang used bidirectional recurrent neural network (RNN) to learn long-term dependency between entity pairs [31]. Furthermore, Zhang et al. proposed bidirectional LSTM network (BLSTM) utilizing position of words, POS tags, named entity information, dependency parse [32]. This model resolved vanishing gradient problem appeared in RNNs by using BLSTM.

Recently, some researcher have proposed attention-based models which can focus to the most important semantic information in a sentence. Zhou et al. combined attention mechanisms with BLSTM [34]. Xiao and Liu split the sentence into two entities and used two attention-based BLSTM hierarchically [21]. Shen and Huang proposed attention-based CNN using word level attention mechanism that is able to better determine which parts of the sentence are more influential [8].

In contrast with end-to-end model, several works proposed models utilizing the shortest dependency path (SDP) between entity pairs of dependency parse trees. SDP-LSTM model proposed by Yan et al. and deep recurrent neural networks (DRNNs) model proposed by Xu et al eliminate irrelevant words out of SDP and use neural network based on the meaningful words composing SDP [24, 23].

3 Model

In this section, we introduce a novel recurrent neural model that incorporate an entity-aware attention mechanism with a LET method in detail. As shown in Fig-

ure 2, our model consists of four main components: (1) **Word Representation** that maps each word in a sentence into vector representations; (2) **Self Attention** that captures the meaning of the correlation between words based on multi-head attention [20]; (3) **BLSTM** which sequentially encodes the representations of self attention layer; (4) **Entity-aware Attention** that calculates attention weights with respect to the entity pairs, word positions relative to these pairs, and their latent types obtained by LET. After that, the features are averaged along the time steps to produce the sentence-level features.

3.1 Word Representation Let a input sentence is denoted by $S = \{w_1, w_2, ..., w_n\}$, where n is the number of words. We transform each word into vector representations by looking up 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 $x_i \in \mathbb{R}^{d_w}$ are 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 sequence of words may expect to learn implicitly of the contextual meaning, but they may not learn well because of the long-term dependency problems [1]. In order for the representation vectors to capture the meaning of words considering the context, we employ the self attention, a special case of attention mechanism, that only requires a single sequence. Self attention has been successfully applied to various NLP tasks such as machine translation, language understanding, and semantic role labeling [20, 17, 19].

We adopt the multi-head attention formulation [20], one of the methods for implementing self attentions. Figure 3 illustrates the multi-head attention mechanism that consists of several linear transformations and scaled dot-product attention corresponding to the center block of the figure. Given a matrix of n vectors, query Q, key K, and value V, the scaled dot-product attention is calculated by the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_w}})V \tag{3.1}$$

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 formulation of multi-head attention is defined by the

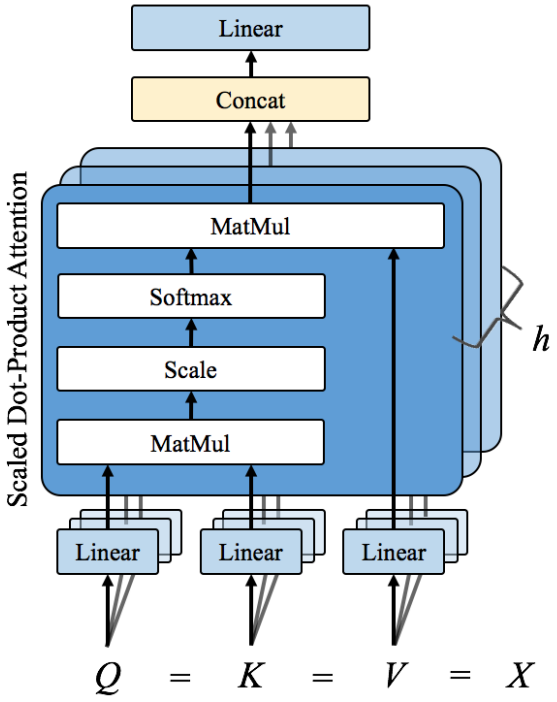

Figure 3: Multi-Head Self Attention. For self attention, the Q(query), K(key), and V(value), inputs of multi-head attention, should be the same vectors. In our work, they are equivalent to X, the word representation vectors.

follows:

$$\text{MultiHead}(Q, K, V) = W^M[head_1; ...; head_r] \tag{3.2}$$

$$head_i = \text{Attention}(W_i^Q Q, W_i^K K, W_i^V V) \tag{3.3}$$

where [;] indicates 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 parameter for linear transformation. W^M is for concatenation outputs of scaled dot-product attention and the others are for query, key, 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 self attention layer is the sequence of representations whose include informative factors in the input sentence.

3.3 Bidirectional LSTM Network For sequentially encoding the output of self attention layer, we use a BLSTM [5, 4] that consists of two sub LSTM networks: a forward LSTM network which encodes the context of a input sentence and a backward LSTM network which encodes that one of the reverse sentence. More formally, BLSTM works as follows:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(m_t) \tag{3.4}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(m_t) \tag{3.5}$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{3.6}$$

The representation vectors M obtained from 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 backward one, where d_h is dimension of each LSTM's state.

$$\overrightarrow{h_t} \in \mathbb{R}^{d_h} \; \overleftarrow{h_t} \in \mathbb{R}^{d_h} \tag{3.7}$$

3.4 Entity-aware Attention Mechanism Although many models with attention mechanism achieved state-of-the-art performance in many NLP tasks. However, for the relation classification task, these models lack of prior knowledge for given entity pairs, which could be powerful hints for solving the task. Relation classification differs from sentence classification in that information about entities is given along with sentences.

We propose a novel entity-aware attention mechanism for fully utilizing informative factors in given entity pairs. Entity-aware attention utilizes the two additional features except $H = \{h_1, h_2, ..., h_n\}$, (1) relative position features, (2) entity features with LET, and the final sentence representation z, result of the attention, is computed as follows:

$$u_i = \tanh(W^H[h_i; p_i^{e_1}; p_i^{e_2}] + W^E[h_{e_1}; t_1; h_{e_2}; t_2]) \tag{3.8}$$

$$\alpha_i = \frac{\exp(v^\top u_i)}{\sum_{j=1}^{n} \exp(v^\top u_j)} \tag{3.9}$$

$$z = \sum_{i=1}^{n} \alpha_i h_i \tag{3.10}$$

3.4.1 Relative Position Features In relation classification, the position of each word relative to entities has been widely used for word representations [30, 14, 8]. Recently, position-aware attention is published as a way to use the relative position features more effectively [33]. It is a variant of attention mechanisms, which use not only outputs of BLSTM but also the relative position features when calculating attention weights.

We adopt this method with slightly modification as shown in Equation 3.8. 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 a index of j-th entity. Similar to word embeddings, the relative positions are converted to vector representations by looking up 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, the representations of BLSTM layer take into account the context and the positional relationship with 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 3.8.

3.4.2 Entity Features with Latent Type Since entity pairs are powerful hints for solving relation classification task, we involve the entity pairs and their types in the attention mechanism to effectively train relations between entity pairs and other words in a sentence. We employ the two entity-aware features. The first is the hidden states of BLSTM corresponding to positions of entity pairs, which are high-level features representing entities. These are denoted by $h_{e_i} \in \mathbb{R}^{2d_h}$, where e_i is index of i-th entity.

In addition, latent types of the entities obtained by LET, our proposed novel method, are the second one. Using types as features can be a great way to improve performance, since the types of entities alone can be inferred the approximate relations. Because the annotated types are not given, we use the latent type representations by applying the LET inspired by latent topic clustering, a method for predicting latent topic of texts in question answering task [26]. The LET constructs the type representations by weighting K latent type vectors based on attention mechanisms. The mathematical formulation is the follows:

$$a_i^j = \frac{\exp((h_{e_j})^\top c_i)}{\sum_{k=1}^{K} \exp((h_{e_j})^\top c_k)} \tag{3.11}$$

$$t_{j\in\{1,2\}} = \sum_{i=1}^{K} a_i^j c_i \tag{3.12}$$

where c_i is the i-th latent type vector and K is the number of latent entity types.

As a result, entity features are constructed by concatenating the hidden states corresponding entity positions and types of entity pairs. After linear transformation of the entity features, they add up with the representations of BLSTM layer as in Equation 3.8, and the representation of sentence $z \in \mathbb{R}^{2d_h}$ is computed by Equations from 3.8 to 3.10.

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 produces the conditional probability $p(y|S, \theta)$ over all relation types:

$$p(y|S, \theta) = \text{softmax}(W^O z + b^O) \tag{3.13}$$

where y is a target relation class and S is the input sentence. The θ is whole learnable parameters in the whole 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:

$$\mathcal{L} = -\sum_{i=1}^{|D|} \log p(y^{(i)}|S^{(i)}, \theta) + \lambda ||\theta||_2^2 \tag{3.14}$$

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 AdaDelta optimizer [29] to compute the parameters θ of our model.

To alleviate overfitting, we constrain the L2 regularization with the coefficient λ [13]. In addition, the dropout method is applied after word embedding, LSTM network, and entity-aware attention to prevent co-adaptation of hidden units by randomly omitting feature detectors [7, 28].

4 Experiments

4.1 Dataset and Evaluation Metrics We evaluate our model on the SemEval-2010 Task 8 dataset, which is an commonly used benchmark for relation classification [6] 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, so the total number of relations is 19. There are 10,717 annotated sentences which consist of 8,000 samples for training and 2,717 samples for testing. We adopt the official evaluation metric of SemEval-2010 Task 8, which is based on the macro-averaged F1-score (excluding *Other*), and takes into consideration the directionality.

4.2 Implementation Details We tune the hyperparameters for our model on the development set randomly sampled 800 sentences for validation. The best hyperparameters in our proposed model are shown in following Table 1.

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}

Table 1: Hyperparameters.

We use pre-trained weights of the publicly available GloVe model [15] to initialize word embeddings in our model, and other weights are randomly initialized from zero-mean Gaussian distribution [3].

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*. First, the SVM [16], *Non-Neural Model*, was top of the SemEval-2010 task, during the official competition period. They used many handcraft feature and SVM classifier. As a result, they achieved an F1-score of 82.2%. The second is *SDP-based Model* such as MVRNN [18], FCM [27], DepNN [9], depLCNN+NS [22], SDP-LSTM [24], and DRNNs [23]. The SDP is reasonable features for detecting semantic structure of sentences. Actually, the SDP-based models show high performance, but SDP may not always be accurate and the parsing time is exponentially increased by long sentences. The last model is *End-to-End Model* automatically learned internal representations can occur between the original

inputs and the final outputs in deep learning. There are CNN-based models such as CNN [30, 14], CR-CNN [2], and Attention-CNN [8] and RNN-based models such as BLSTM [32], Attention-BLSTM [34], and Hierarchical-BLSTM (Hier-BLSTM) [25] for this task.

	Model	F1
Non-Neural Model	SVM	82.2
SDP-based Model	MVRNN	82.4
	FCM	83.0
	DepNN	83.6
	depLCNN+NS	**85.6**
	SDP-LSTM	83.7
	DRNNs	**86.1**
End-to-End Model	CNN	82.7
	CR-CNN	84.1
	Attention-CNN	84.3
	+ POS, WN, WAN	**85.9**
	BLSTM	82.7
	+ PF, POS, NER, DEP, WN	84.3
	Attention-BLSTM	84.0
	Hier-BLSTM	84.3
	Our Model	**84.7**
	+ Latent Entity Typing	**85.2**

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

Our proposed model achieves an F1-score of 85.2% which outperforms all competing state-of-the-art approaches except depLCNN+NS, DRNNs, and Attention-CNN. However, they rely on high-level lexical features such as WordNet, dependency parse trees, POS tags, and NER tags from NLP tools.

The experimental results show that the LET is effective for relation classification. The LET improve a performance of 0.5% than the model not applied it. The model showed the best performance with three types.

5 Visualization

There are three different visualization to demonstrate that our model is more interpretable. First, the visualization of self attention shows where each word focus on parts of a sentence. By showing the words that the entity pair attends, we can find the words that well represent the relation between them. Next, the entity-aware attention visualization shows where the model pays attend to a sentence. This visualization result highlights important words in a sentence, which are usually important keywords for classification. Finally, we visualize representation of type in LET by using t-SNE [10], a method for dimensionality reduction, and group the whole entities in the dataset by the its latent types.

5.1 Self Attention We can obtain the richer word representations by using self attentions. These word representations are considered the context based on correlation between words in a sentence. The Figure 4 illustrates the results of the self attention in the sentence, "*the ⟨e1⟩pollution⟨/e1⟩was caused by the ⟨e2⟩shipwrek⟨/e2⟩*", which is labeled *Cause-Effect(e1,e2)*. There are visualizations of the two heads in the multi-head attention applied for self attention. The color density indicates the attention values, results of Equation 3.1, which means how much an entity focuses on each word in a sentence.

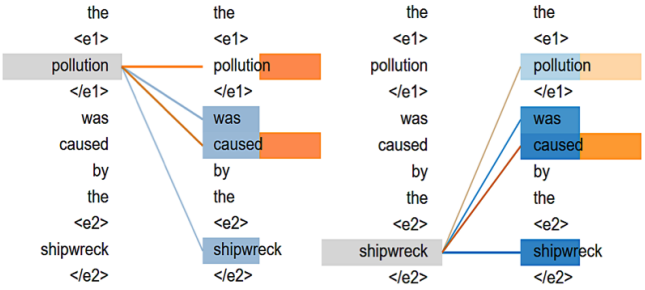

Figure 4: Visualization of Self Attention.

In Figure 4, the left 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(e1,e2)*, which is the relation class of this entity pair.

5.2 Entity-aware Attention Figure 5 shows where the model focuses on the sentence to compute relations between entity pairs, which is the result of visualizing the alpha vectors in Equation 3.9. The important words in sentence are highlighted in yellow, which means that the more clearly the color is, the more important it is. For example, in the first sentence, the *inside* is strongly highlighted, which is actually the best word representing the relation *Component-whole(e1,e2)* between the given entity pair. As another example, in the third sentence, the highlighted *assess* and *using* represent the relation,

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 5: Visualization of Entity-aware Attention

Instrument-Agency(e2,e1) between entity pair, *analysts* and *frequency*, well. We can see that the *using* is more highlighted than the *assess*, because the former represents the relation better.

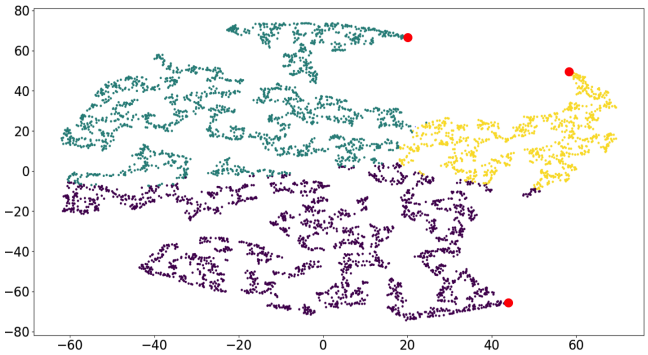

Figure 6: Visualization of latent type representations using t-SNE

5.3 Latent Entity Type Figure 6 visualizes latent type representation $t_{j\in\{1,2\}}$ in Equation 3.12 Since the dimensionality of representation vectors are too large to visualize, we applied the t-SNE, one of the most popular dimensionality reduction methods. In Figure 6, the red points represent latent type vectors $c_{i\in K}$ and the rests are latent type representations t_j, where the colors of points are determined by the closest of the latent type vectors in the vector space of the original dimensionality. The points are generally well divided and are almost uniformly distributed without being biased to one side.

Figure 7 summarizes the results of extracting 50 entities in close order with each latent type vector. This allows us to roughly understand what latent types of entities are. We use a total of three types and find that similar characteristics appear in words grouped by together. In the type 1, the words are related to human's jobs and foods. The type2 has a lot of entities related to machines and engineering like *engine*, *woofer*, and *motor*. Finally, in type3, there are many words with bad meanings related associated with disasters and drugs. As a result, each type has a set of words with similar characteristics, which can prove that LET works effectively.

Type1 : *worker, chairman, author, king, potter, cuisine, spaghetti, restaurant, sugars, bananas, salad, bean*
Type2 : *systems, engine, trucks, valve, hinge, assembly, woofer, mainspring, wriggle, circuit, motor*
Type3 : *virus, tsunami, accident, dust, riot, pandemic, pollution, earthquake, contamination, debt,, congestion, drugs, marijuana*

Figure 7: Sets of Entities grouped by Latent Types

6 Conclusion

In this paper, we proposed 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 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, our three visualizations of attention mechanisms applied to the model demonstrate that our model is more interpretable than previous models. We expect our model to be extended not only the relation classification task but also other tasks that entity plays an important role. Especially, latent entity typing can be effectively applied to sequence modeling task using entity information without NER. In the future, we will propose a new method in question answering or knowledge base population based on relations between entities extracted from our model.

References

[1] Y. Bengio, P. Simard, and P. Frasconi, *Learning long-term dependencies with gradient descent is difficult*, IEEE transactions on neural networks, 5 (1994), pp. 157–166.

[2] C. dos Santos, B. Xiang, and B. Zhou, *Classifying relations by ranking with convolutional neural networks*, in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol. 1, 2015, pp. 626–634.

[3] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010, pp. 249–256.

[4] A. Graves, A.-r. Mohamed, and G. Hinton, *Speech recognition with deep recurrent neural networks*, in Acoustics, speech and signal processing (icassp), 2013 ieee international conference on, IEEE, 2013, pp. 6645–6649.

[5] A. Graves and J. Schmidhuber, *Framewise phoneme classification with bidirectional lstm and other neural network architectures*, Neural Networks, 18 (2005), pp. 602–610.

[6] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, *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, Association for Computational Linguistics, 2009, pp. 94–99.

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors*, arXiv preprint arXiv:1207.0580, (2012).

[8] X. Huang et al., *Attention-based convolutional neural network for semantic relation extraction*, in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 2526–2536.

[9] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, and H. Wang, *A dependency-based neural network for relation classification*, arXiv preprint arXiv:1507.04646, (2015).

[10] L. v. d. Maaten and G. Hinton, *Visualizing data using t-sne*, Journal of machine learning research, 9 (2008), pp. 2579–2605.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, arXiv preprint arXiv:1301.3781, (2013).

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality*, in Advances in neural information processing systems, 2013, pp. 3111–3119.

[13] A. Y. Ng, *Feature selection, l 1 vs. l 2 regularization, and rotational invariance*, in Proceedings of the twenty-first international conference on Machine learning, ACM, 2004, p. 78.

[14] T. H. Nguyen and R. Grishman, *Relation extraction: Perspective from convolutional neural networks*, in Proceedings of the NAACL Workshop on Vector Space Modeling for Natural Language Processing, 2015, pp. 39–48.

[15] J. Pennington, R. Socher, and C. Manning, *Glove: Global vectors for word representation*, in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[16] B. Rink and S. Harabagiu, *Utd: Classifying semantic relations by combining lexical and semantic resources*, in Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, 2010, pp. 256–259.

[17] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang, *Disan: Directional self-attention network for rnn/cnn-free language understanding*, arXiv preprint arXiv:1709.04696, (2017).

[18] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, *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, Association for Computational Linguistics, 2012, pp. 1201–1211.

[19] Z. Tan, M. Wang, J. Xie, Y. Chen, and X. Shi, *Deep semantic role labeling with self-attention*, arXiv preprint arXiv:1712.01586, (2017).

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[21] M. Xiao and C. Liu, *Semantic relation classification via hierarchical recurrent neural network with attention*, in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 1254–1263.

[22] K. Xu, Y. Feng, S. Huang, and D. Zhao, *Semantic relation classification via convolutional neural networks with simple negative sampling*, arXiv preprint arXiv:1506.07650, (2015).

[23] Y. Xu, R. Jia, L. Mou, G. Li, Y. Chen, Y. Lu, and Z. Jin, *Improved relation classification by deep recurrent neural networks with data augmentation*, arXiv preprint arXiv:1601.03651, (2016).

[24] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, *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, 2015, pp. 1785–1794.

[25] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, *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, 2016, pp. 1480–1489.

[26] S. Yoon, J. Shin, and K. Jung, *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, Volume 1 (Long Papers), Association for Computational Linguistics, 2018, pp. 1575–1584.

[27] M. Yu, M. Gormley, and M. Dredze, *Factor-based compositional embedding models*, in NIPS Workshop on Learning Semantics, 2014, pp. 95–101.

[28] W. Zaremba, I. Sutskever, and O. Vinyals, *Recurrent neural network regularization*, arXiv preprint arXiv:1409.2329, (2014).

[29] M. D. Zeiler, *Adadelta: an adaptive learning rate method*, arXiv preprint arXiv:1212.5701, (2012).

[30] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, *Relation classification via convolutional deep neural network*, in Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, 2014, pp. 2335–2344.

[31] D. Zhang and D. Wang, *Relation classification via recurrent neural network*, arXiv preprint arXiv:1508.01006, (2015).

[32] S. Zhang, D. Zheng, X. Hu, and M. Yang, *Bidirectional long short-term memory networks for relation classification*, in Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation, 2015, pp. 73–78.

[33] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, *Position-aware attention and supervised data improve slot filling*, in Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 35–45.

[34] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, *Attention-based bidirectional long short-term memory networks for relation classification*, in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), vol. 2, 2016, pp. 207–212.