Article

Neural Relation Classification Using Selective Attention and Symmetrical Directional Instances

Zhen Tan * , Bo Li, Peixin Huang, Bin Ge and Weidong Xiao

Science and Technology on Information Systems Engineering Laboratory,
National University of Defense Technology, Changsha 410073, China; libo11@nudt.edu.cn (B.L.);
huangpeixin15@nudt.edu.cn (P.H.); gebin@nudt.edu.cn (B.G.); wdxiao@nudt.edu.cn (W.X.)

* Correspondence: tanzhen08a@nudt.edu.cn; Tel.: +86-151-1144-0303

Received: 3 July 2018; Accepted: 19 August 2018; Published: 21 August 2018

Abstract: Relation classification (RC) is an important task in information extraction from unstructured text. Recently, several neural methods based on various network architectures have been adopted for the task of RC. Among them, convolution neural network (CNN)-based models stand out due to their simple structure, low model complexity and “good” performance. Nevertheless, there are still at least two limitations associated with existing CNN-based RC models. First, when handling samples with long distances between entities, they fail to extract effective features, even obtaining disturbing ones from the clauses, which results in decreased accuracy. Second, existing RC models tend to produce inconsistent results when fed with forward and backward instances of an identical sample. Therefore, we present a novel CNN-based sentence encoder with selective attention by leveraging the shortest dependency paths, and devise a classification framework using symmetrical directional—forward and backward—instances via information fusion. Comprehensive experiments verify the superior performance of the proposed RC model over mainstream competitors without additional artificial features.

Keywords: relation extraction; select attention; symmetrical directional instances; information fusion

1. Introduction

The task of relation classification (RC) is to determine the type of semantic relation between two entities from unstructured text, which is considered to be of significance in various natural language processing (NLP) applications. That is, RC is incorporated as an intermediate step in many complex NLP applications, e.g., information extraction, automatic knowledge base construction, etc.

For example, given a sample sentence “Financial stress is one of the main causes of divorce.”, there are two mentions of target entities marked by e_1 = “stress” and e_2 = “divorce”. The goal of RC is to recognize the semantic relation of cause-effect automatically between the entities e_1 and e_2 from that sentence.

In practice, the expressions of a particular relation, i.e., mentions of relation, can be in various forms, in terms of words, syntax, as well as context. As a consequence, such a phenomenon poses a serious technical challenge to performing accurate RC. With the recent advance in neural networks, various models were proposed to learn syntactic features from raw sentences or parse trees [1,2]. Convolution neural network (CNN) models leverage multi-layer convolution kernels to extract high-level features, which can achieve well-matched performance with other RC models, by utilizing a standard pattern with tri-part components: a convolution layer, a pooling layer and a softmax layer. Albeit that they provide “good” performance, we observe that they are associated with at least two limitations.

In the aforementioned example, the two target entities are quite close to each other, and the desired target relation is easy to identify; in the wild, however, there are much more cases where longer distances exist between entities, and in this case, CNN may fail to extract effective features, or even elicits erroneous features from those irrelevant parts—subsequences and clauses—of the sentence. Let us take the following sentence as an example: “We poured the milk, which is made in China, into the mixture”; “made in” closely relates to the relation product-producer, while “pour” and “into” relate to relation entity-destination. CNN may classify it as product-producer by the high level feature of “made”, but the desired relation is entity-destination. Furthermore, it is observed that existing methods produce inconsistent results when fed with forward and backward instances (to be formally defined in Section 3.2) of the same sentence. For the sample sentence in the beginning, the classifier may get cause-effect if treating “stress” as e_1 and “divorce” as e_2 (forward instance), but get component-whole if taking “divorce” as e_1 and “stress” as e_2 (backward instance). Attributing to this overlooked fact, the data modeling of existing methods could be potentially refined for better performance.

This article addresses the aforementioned challenges, and the main contributions are at least three-fold:

- * For the first limitation, we propose a CNN-based sentence encoder integrated with a selective attention layer, which leverages the shortest dependency path to help find keywords closely related to desired relations.
- * For the second limitation, we reconstruct the multi-classification framework via information fusion for accurate RC, such that symmetrical directional instances are consolidated for data argumentation.
- * The proposed techniques constitute a novel method, and comprehensive experimental study indicates that the proposed model achieves state-of-the-art results on F1 score over the SemEval-2010 Task 8 dataset.

Organization: In Section 2, we survey related work from existing literature. We present the new model in Section 3, followed by experiments in Section 4. We conclude the article in Section 5.

2. Related Work

Thus far, conventional approaches to RC can be put into two categories, i.e., feature-based methods and kernel-based methods. Feature-based models rely on various kinds of human-engineered or hand-crafted features, while kernel-based models leverage various kernels. Feature-based methods leverage various hand-designed features, firstly to represent the latent syntactic and semantic cues in each sentence; and then, one or more relation classifiers, e.g., Support Vector Machine (SVM), and the combined hand-designed features are utilized to judge the relation type of each sentence [1,3,4]; thus, they cost a large amount of time to construct features and are hard to apply on large-scale RC tasks. Tree kernel-based methods project the features into a high-dimensional space, and the inner products are leveraged to calculate the similarity between different structural features. Zelenko et al. transferred each sentence into a kernel tree which, consisted of various weighted common subtrees and could be utilized to capture the commonality of each shallow parse tree [5,6]. Culotta and Sorensen leveraged the dependency tree to design the kernel features, and each node in the dependency tree included many other syntactic features, e.g., POS (part-of-speech) tag, word chunk tag, and so forth [7]. Zhou et al. leveraged content-based and semantic features to construct a convolution kernel tree, which can extract model semantic features in each sentence [8,9].

Nonetheless, the aforementioned methods suffer from the serious issue of error propagation and, hence, exhibit poor capability of generalization on unseen text. Recently, neural network-based methods [10,11] have been designed to address the problem, mainly including CNN [2,12–16] and recurrent neural network (RNN)-based [17–22] methods. These methods [12,23,24] leverage the embedding to represent each token in the sentence, which capture the latent semantic features more

effectively and overcome the semantic representation problem in feature-based and kernel-based methods. In particular, Zeng et al. fed various latent lexical and semantic features over the whole sentence to a CNN to extract the relation features [2]. Besides, the position information of each token was utilized to extract the latent features in each sentence, which can also improve the performance of RC effectively. Santos et al. tackled the problem by using a CNN that performed RC by ranking [13]. Socher et al. proposed a recursive neural network model to capture the relation features, which leveraged the constituent parse tree and correlation between two entities to construct the input embeddings [25].

Besides embeddings of raw words as used in the methods above, research from [17,26] exploited the shortest dependency path (SDP) as the input of neural networks. Distinctively, we introduce SDP to generate a selective attention matrix, which has not been explored by existing methods. In addition, we are aware of a few improved models, including ATT-BiLSTM [19] and Bi-LSTM-RNN [22]. Notably, Nguyen and Grishman proposed to combine a recurrent neural network and CNN and utilized an explicit voting-based method to achieve RC [18]. Concerning RC, however, RNN and LSTM are not as effective as they are on other NLP tasks. Among others, we are aware of a few attention-augmented methods [16,19,27] for neural RC. These attention mechanisms are sophisticated, and the performance gain may not always match the payout.

3. Proposed Method

Given a sentence $x = [x_1, x_2, \dots, x_n]$, there are 2 annotated target entities, denoted by e_1 and e_2 (without loss of generality, it is assumed that every entity corresponds to exactly one word in the sentence, namely the entity word, and a target entity may be instantiated by more than one word, in which case, the position of the first word is used for distance evaluation), where x_i is a word of the sentence x , $i \in [1, n]$, and the task of RC is to determine the type of semantic relation, denoted by r_i , between the two annotated entities e_1 and e_2 from a set of candidate relations, denoted by $R = \{r_1, r_2, \dots, r_m\}$, $m = |R|$.

We introduce the proposed model with a bottom-up design consideration. We first introduce the core components of the sentence encoder and then present the classification framework using symmetrical directional instances via information fusion.

3.1. Sentence Encoder

Given a sentence, a CNN with selective attention, coined SA-CNN, designed as the sentence encoder to construct a low-dimensional vector of real values to represent the sentence. Firstly, every word in the sentence is transformed into a dense feature vector of real values, and the convolution layer is used to extract high-level features of the sentence. Then, we incorporate a selective attention layer to improve the focus of the sentence encoder on crucial words with respect to the desired semantic relation. Lastly, the max-pooling layer and non-linear layer are used to construct a distributed representation of the sentence. We sketch the aforementioned model of SA-CNN in Figure 1.

Input representation and convolution layer: Firstly, at the layer of representation of input sentences, we mainly transform the words of sentences, through embedding techniques, to low dimensional vectors. The original inputs of SA-CNN are raw words of the sentence x . Since CNN only handles input sentences of a fixed length, before sending the sentences into CNN, we employ the conventional padding scheme, which converts the original sentences to make up sequences of an identical length. The target length is set to, among all the sentences in the corpus, that of the longest sentence, denoted by n , and "NAN" is used by default for this purpose.

Besides classic word embeddings, in order to keep a record of the positions of the annotated entities, we append a position embedding to each word. Moreover, to enhance the comprehension and incorporation of sentence-dependent structures, we also supplement a dependency direction

embedding and a dependency tag embedding for the aforementioned purpose. We briefly describe each of them below.

Word embeddings: Word embeddings leverage vectors to represent corresponding words and are designed to capture the syntactic and grammatical features of each word, simultaneously. Given a sentence of n words $x = [x_1, x_2, \dots, x_n]$, x_i represents the i -th word in the sequence, $i \in [1, n]$, and n is the padding length. Before sending them to the network, we look up, for each word x_i , its distributed representation e_i , i.e., word embedding vector, from a word embedding table W , in which the dimensionality of word embeddings is m_e . In the experiment, we used a pre-trained W .

Figure 1. Convolution neural network (CNN) sentence encoder with selective attention.

Position embeddings: In RC, the words that appear around the target entities tend to be helpful to determine the semantic relation between the entities. In addition, position embedding is necessary, as CNN loses the positional information of annotated entities in the sentence if position information is not provided. As a consequence, we choose to replenish position embeddings to the basic word embeddings, which are specified in terms of entity pairs.

Definition 1 (Word distance). *Given a sentence $x = [x_1, \dots, x_i, \dots, x_j, \dots, x_n]$, $1 \leq i < j \leq n$, $n = |x|$, the distance from word x_i (resp. x_j) to word x_j (resp. x_i) equals $i - j$ (resp. $j - i$).*

For each word x_i , the distance of x_i from itself to the annotated entity word x_{i_1} (resp. x_{i_2}) in the sentence is $i - i_1$ (resp. $i - i_2$), respectively, which is then transformed to a vector of real values d_{i_1} (resp. d_{i_2}). These vectors are the position embeddings taken from a position embedding table D , which is initialized randomly. Note that the distances range from $1 - n$ to $n - 1$, and hence, the position embedding table is of size $(2n - 1) \cdot m_p$, where m_p is the dimensionality of position embeddings.

Dependency embeddings. Dependency structures of sentences play an important role in RC, which provide syntax relationships among the words. Nevertheless, CNN processes high-level features among words inside a sliding window. In this connection, we further design dependency embeddings to assist CNN in understanding the dependency relationship in a small range within the CNN window. In other words, for every word in the sentences, we further append 2 dependency embeddings, which are constituted of a direction embedding and a tag embedding.

In particular, we employ the dependency-based parse tree for deriving dependency embeddings. The dependency-based parse tree is a tree composed of interdependence between words. We use an example to illustrate it (Figure 2). In the sample dependency-based parse tree, there are dependencies

among the words in the sentence, indicated by arcs (from lower-level to upper-level) each bearing a dependency tag.

Specifically, for every word x_i , to obtain the direction embedding, we compute the distance from x_i to the word in the upper-level of the dependency-based parse tree; for tag embedding, the arc tag from x_i to the word in the upper-level is utilized. Take the sentence in Figure 2 for instance. The distance from “broke” to “thief”, which is in the upper-level, is 8, and the corresponding tag is “nsubj”. Then, for each word x_i , its distances to the word in the upper-level and the corresponding tag are, respectively, converted into vectors of real values p_i and f_i . Akin to the aforementioned mapping procedure, we use a dependency direction embedding table P and a dependency tag embedding table F for the purpose of fast mapping, which are randomly initialized. The dimensionalities of direction and tag embedding are m_d and m_f , respectively.

Figure 2. An example of a dependency-based parse tree.

As depicted in Figure 1, the word embeddings e_i , the position embeddings d_{i_1} and d_{i_2} and the dependency embeddings p_i and f_i are concatenated together into one vector of real values. That is, $X_i = [e_i, d_{i_1}, d_{i_2}, p_i, f_i]^T$, which is hence used to represent x_i . By doing this, the original input sentence x is now transformed into a real-valued matrix $X = [X_1, X_2, \dots, X_n]$ of size $n \cdot (m_e + 2m_p + m_d + m_f)$.

Convolution: In RC, input sentences can be of variable lengths, and critical information for determining semantic relations may appear in any part of the sentence. This inspires us to utilize all local features while performing relation prediction in a global manner. In this connection, we employ a convolution layer to mix and digest all the local features.

Firstly, the convolution layer utilizes a w -length sliding window to extract the local latent features in each sentence. In the example shown in Figure 1, we assume that the size of the sliding window w is 3. In order to convolute the word with same probability, we add the padding tokens in the start and end positions of each sentence. This means that we regard all the input vectors x_i as out-of-range, i.e., $i < 1$ or $i > n$, as zero vectors. Specifically, the convolution filter is seen as a weight matrix $f = [f_1, f_2, \dots, f_w]$, where f_i is a column vector of size $m_e + 2m_p + m_d + m_f$.

The core of this convolution layer is derived from the application of the convolution operator on the two matrices X and f , which produces a score sequence, denoted by $s = [s_1, s_2, \dots, s_n]$, where:

$$s_i = g\left(\sum_{j=0}^{w-1} f_{j+1}^T X_{j+1}^T + b\right),$$

where b is a bias and g is a non-linear activation function. Similar to the classic CNN model, this convolution process may be replicated multiple times using different filters with different window sizes.

Selective attention and max-pooling layer: During convolution, CNN does not discriminate the words, whereas treating each word to have equal contribution to the desired semantic relation; intuitively, however, in a sentence, every word has different significance to defining the relationship between target entities. We evaluate such significance by the shortest dependency path (SDP).

Definition 2 (Shortest dependency path). *Given a sentence and its dependency-based parse tree, the shortest dependency path between target entities is the shortest undirected path between the entity words in the dependency-based parse tree.*

Take the following sentence for instance, “A thief, who intends to go to the city, broke the ignition with screwdriver.”, the dependency-based parse tree of which is in Figure 2. The SDP between “thief” and “screwdriver” is “thief—nsubj—broke—nmod—screwdriver”. The semantic relation between “thief” and “screwdriver” is instrument-agency, and the keyword “broke” is closely related to the relation, while the word “go” relates to entity-destination. Without considering SDP, it is likely to classify it as entity-destination, which is wrong.

Specifically, keywords are defined as those on the SDP between entity words, which are deemed to be closely related to the semantic relation of the sentence, and are able to improve the focus of the sentence encoder through the selective attention layer. Thus, we add a selective attention layer after the convolution layer (while other sophisticated attention mechanisms are available, e.g., [16,19,27], and can be incorporated, we show that such a simple proposal works extraordinarily well), such that the crucial words are more weighted to elevate the attention of the sentence encoder. Since the semantic relation of a sentence is usually determined by a sequence of words, rather than the keyword only, we also weight the words around the keywords.

Intuitively, the closer a word is to the keywords, the greater the weight it receives. To this end, we set two coefficients: weight coefficient α ($\alpha > 1$) and distance damping factor β ($0 < \beta < 1$). Hence, for each word x_i , the weight q_i is determined by distance d_i , which is the minimum of the shortest distances from x_i to the keywords. Thus, $q_i = \alpha \cdot \beta^{d_i}$, and the selective attention weight matrix is $M_A = \text{diag}([q_1, q_2, \dots, q_n])$, where diag is a diagonal matrix. Hence, the score matrix $s_A = M_A \cdot s$.

In the pooling layer, we use a max function to obtain the most important features. For each filter block, the feature with maximum value is selected as output, i.e., $p_f = \max\{s_A\}$, $f \in [1, t]$, where t is the number of convolution filters. Afterwards, we concatenate all the scores from the filters to represent the sentence as $z = [p_1, p_2, \dots, p_t]$.

As the last step, we apply a non-linear activation function on top of z as the feature vector of the sentence, or equally sentence embedding, with which a conventional multi-layer perceptron (MLP) with a softmax layer is able to classify the semantic relation of the sentence.

3.2. Classification Framework

With the sentence embedding, while an MLP with a softmax layer works, the accuracy is of particular interest. In a thorough investigation, we observe that for one sentence, different orders of target entities may introduce inconsistent classification results.

For instance, “Financial stress is one of the main causes of divorce.”, the MLP-based classifier identifies the semantic relation between “stress” and “divorce” as cause-effect, but puts the relations between “divorce” and “stress” into entity-destination, which should be effect-cause. This implies that the basic MLP-based classification framework may not model the sentences well and, thus, gives rise to inconsistent results. Therefore, we propose to jointly consider the two situations to enhance the accuracy and develop the symmetrical directional-instance classification framework (DI), which is depicted in Figure 3.

Particularly, we define the symmetrical directional instance for target entities.

Definition 3. Consider a sentence $x = [x_1, \dots, x_i, \dots, x_j, \dots, x_n]$, $1 \leq i < j \leq n$, with two target entities e_1 and e_2 corresponding to x_i and x_j , respectively. The forward instance of e_1 and e_2 is defined as (e_1, e_2) , and the backward instance is defined as (e_2, e_1) .

In essence, the forward instance reflects the literal order of the target entities, and the backward instance entails the opposite. The backward instance provides additional information for identifying the desired semantic relation, and thus, the information from both directional instances needs to be fused for more accurate classification. This recalls two classic strategies of information fusion for decision-making: (1) feature-in-decision-out (FEI-DEO), i.e., the inputs are features, whereas the

outputs are decisions; and (2) decision-in-decision-out (DEI-DEO), i.e., decisions are fused to obtain enhanced or new decisions.

Figure 3. Classification with symmetrical directional instances. FEI-DEO, feature-in-decision-out; DEI-DEO, decision-in-decision-out.

As depicted in Figure 3, for the FEI-DEO scenario, the embeddings of both directional instances are fused first into one feature vector and passed over for classification; for the DEI-DEO scenario, the embeddings of symmetrical directional instances are sent to the softmax layer separately, generating two classification vectors, which are fused for the final result. For both scenarios, we use a softmax layer as the classifier, and two types of fusion techniques are tested: weighted average and artificial neural network (ANN). Therefore, four kinds of methods in total are evaluated in Section 4. For DEI-DEO with a weighted average, we redefine the goal function, as the forward relation may have a conflict with the backward one. As a consequence, we incorporate as a weighting vector the parameter ω to combine the likelihood of the forward and the backward symmetrical instances. The loss function is:

$$J(\theta) = \sum_{i=1}^n \omega (\log p(r_i^+ | z_i^+, \theta)) + (1 - \omega) (\log p(r_i^- | z_i^-, \theta')),$$

where n is the number of sentences and θ and θ' are the two sets of parameters, respectively, in the model using forward and backward instances. The superscripts $+$ and $-$ denote the positive and negative samples, respectively. $\omega = \frac{1}{1 + e^{-\sigma r_i}}$ is a trade-off weight between the probability of z_i^+ being r_i^+ and the probability of z_i^- being r_i^- .

In the test, we get the classification probability vector of forward instance $C^+ = [c_1, c_2, \dots, c_r]$ and backward instance $C^- = [c_1, c_2, \dots, c_r]$, where c_i is the probability that there is relation r_i between e_1 and e_2 . The final classification is:

$$C = \omega C^+ + (1 - \omega) C^-.$$

Then, we use the function argmax to get the right relation r_i .

For the other three kinds of methods tested, they all produced one relation likelihood distribution. The loss function is:

$$J(\theta) = \sum_{i=1}^n \log p(r_i | z_i, \theta).$$

We optimize the objective functions via Adadelata, which reduces the step size as the number of iterations increases [28], and the training sample order is inconsistent in different iterations. The optimized parameters are $\xi = 10^{-1}$, $\rho = 0.95$ for σ and $\xi = 10^{-6}$, $\rho = 0.95$ for θ and θ' .

4. Experiment

The experimental study is to verify that: (1) the SA-CNN sentence encoder can learn good feature representation from a long sentence; and (2) the DI classification framework effectively improves performance using symmetrical directional instances.

We conducted experiments on the benchmark dataset of SemEval-2010 Task 8, which contains 10,717 instances: 8000 for training and 2717 for testing. It consists of sentences manually labeled with 19 relations (9 directed relations with its inverse and 1 artificial class “Other”). The official scoring script and report the macro F1 score are leveraged to evaluate the performance of our model.

4.1. Baselines

We compare SA-CNN with 8 state-of-the-art RC models. The details are as follows:

- * SVM [1] leverages hand-designed features to describe sentence features and then uses the SVM classifier to judge the relation type between different entities.
- * FCM [24] utilizes word embedding, dependency parse, NERtools and multi-layer perceptron (MLP) to extract latent features, and then, a softmax layer is leveraged to predict the output relation label.
- * CNN [2] is a simple and effective model that comprises a standard convolution layer with filters of four window sizes, followed by a softmax layer for classification.
- * CR-CNN [13] is an improved model that designs a ranking-based classifier with a novel pairwise loss function to reduce the impact of “Other” classes.
- * depLCNN+ NS [14] proposes a straightforward negative sampling to reduce irrelevant information introduced when subjects and objects are at a long distance.
- * Bi-LSTM-RNN [22] utilizes a bi-directional long-short-term-memory recurrent-neural network (Bi-LSTM-RNN) model based on low-cost sequence features to extract latent features where the features are divided into five parts: two entities and their three contexts.
- * RNN [21] leverages a hierarchical recurrent neural network with attention mechanism to extract the latent features between entities, and a softmax function is used to predict the relation type.
- * ATT-Bi-LSTM [19] leverages attention-based Bi-LSTM to capture semantic information and judge the importance of each position in a sentence.

4.2. Analysis of the Proposed Method

We designed two experiments to evaluate the proposed techniques.

Effectiveness of the sentence encoder. In this experiment, symmetrical directional instances were not utilized. On the basis of the baseline system, we added the dependency embeddings, after which it increased the F1 score from 82.1–82.6; next, we further incorporated the selective attention layer, and finally, the performance was improved to 84.1.

This experiment shows that by injecting dependency embeddings, the model is able to extract high-level features based on dependency information, which improves the F1 score. Furthermore, we found that although the dependency embeddings have encoded all the dependency information, the improvement was not remarkable. Only when the distance from a word to its parent in dependency-based parse tree was small enough, i.e., within the sliding window, could CNN extract the dependency structure inside the phrase/sentences.

Furthermore, by adding a selective attention layer, the model exerted selective attention over keywords, which improved the performance significantly. The baseline system may extract the feature of words that are not closely related to target entities, but semantically related to the desired relation.

For instance, it classified a sentence containing “cause” in a clause to cause-effect because of “cause”, which resulted in inaccuracy. Superior to the baseline, SA-CNN took the semantically important words into account and, hence, reduced the chance of erroneous classification.

To prove that SA-CNN was more capable of handling sentences with far-apart target entities, we compared classification accuracy against the distance between target entities, with results plotted in Figure 4. Note that there were few sentences longer than 15 words, and thus, we took the maximum length as 15.

Figure 4. Accuracy versus entity distance distribution.

From the figure, we have the following findings: (1) When the distance is beyond 5, as it increases, the performance of both systems drops in a notable manner. This implies that distance between target entities significantly influences accuracy. (2) In comparison with the baseline, the proposed system effectively improves the performance, especially on long sentences with distance between 6 and 12. The margin is as large as 8.7 when the distance is 10. (3) This brings to attention that the F1 score surges to 83.2 when the distance is 14. We argue that this is not representative, since there are only 9 sentences of this category. When the distance goes beyond 14, the improvement is not obvious, as dependency analysis of very long sentences is fairly difficult, leading to futile selective attention.

Effectiveness of symmetrical directional instances: In this experiment, the basic CNN sentence encoder was used. First, we tested the FEI-DEO framework and observed that both weighted average and ANN methods obtained F1 scores a little lower than that of the baseline. As the embeddings of forward and backward instances were similar, fusion increased the model complexity, resulting in poor performance on this small dataset. Then, we tested the DEI-DEO framework, and the weighted average method obtained an F1 score of 83.5, while the ANN method obtained 82.4. The enhancement indicates that: (1) the fusion of symmetrical directional instances reduces the appearance of erroneous classification; and (2) the root cause can be attributed to the ensemble of two classifiers, trained from different aspects of the same samples, in which case, the overall size of training data is “doubled” to 16,000.

We also conducted the experiment of feeding all 16,000 instances to only one classifier, and the result was roughly the same as throwing dice. On the one hand, such a result backs up the second argument above; on the other hand, this is ascribable to the resemblance of embeddings of symmetrical directional instances, but the disparity of semantic relations behind, which confuses the classifier.

4.3. Evaluation against State-of-the-Art Competitors

This experiment compares the proposed method, denoted by SA-CNN + DI, with state-of-the-art methods, and the results are summarized in Table 1. Among the 3 classes—conventional, CNN-based

and RNN-based methods—CNN-based methods take the most ascendant position. The comparison with conventional methods demonstrates that human-designed features cannot concisely express the semantic meaning of the sentences. By integrating together all the proposed techniques, SA-CNN + DI demonstrated the best result of F1 score, 85.8, almost 4% superior to the SVM-based method. We also remark that the basic CNN-based method achieved a fairly good initial result with a very simple model structure, and we chose CNN as the basis and further enhanced it by adding some delicate components, while still keeping the model simple.

Table 1. Experiment results of the comparison. SA, selective attention.

Methods	Additional Features	F1
SVM [1]	POS, prefixes, morphological, WordNet, Levin classed, ProBank, FramNet, NomLex-Plus, Google <i>n</i> -gram, paraphrases, TextRunner	82.2
FCM [24]	word, dependency parsing, NER	83.0
CNN [2]	words around entities, WordNet	82.7
CR-CNN [13]	words, word position	84.1
depLCNN+ NS [14]	WordNet, words around entities	85.6
RNN [21]	words, position indicators	79.6
Bi-LSTM-RNN [22]	word, char, POS, WordNet, dependency, POS	83.1
ATT-Bi-LSTM [19]	WordNet, grammar	83.7
SA-CNN + DI	words, dependency	85.8

5. Conclusions

In this paper, we have unveiled two issues of neural relation classification when dealing with far apart entities, as well as forward and backward instances in a single sentence, which was overlooked by existing literature. We proposed a novel method to address the challenges. The method utilizes a CNN with selective attention as the sentence encoder, in which the shortest dependency paths are exploited for identifying keywords. The sentence vectors are input into a classification framework inspired by information fusion over symmetrical directional instances. Experiment results demonstrated the merits of the proposed method: a simple model, yet have superior performance.

In future work, we want to fuse more internal and external features to improve the performance, e.g., the entity description information, the semantic correlation between different entities, and so forth. Besides, we plan to explore the potential application of the proposed model and techniques in other related tasks, such as knowledge graph construction and joint entity and relation extraction.

Author Contributions: Conceptualization, Z.T. and B.L.; Methodology, B.L.; Software, Z.T. and B.L.; Validation, Z.T.; Formal Analysis, Z.T.; Investigation, B.G.; Resources, Z.T. and B.L.; Data Curation, Z.T. and B.L.; Writing—Original Draft Preparation, P.H.; Writing—Review & Editing, Z.T., P.H. and B.L.; Visualization, B.L.; Supervision, W.X.; Project Administration, W.X.; Funding Acquisition, W.X.

Funding: This research was funded by NSFC grant number 61872466, 71690233, 71331008.

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

References

1. Rink, B.; Harabagiu, S.M. UTD: Classifying Semantic Relations by Combining Lexical and Semantic Resources. In Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala University, Uppsala, Sweden, 15–16 July 2010; pp. 256–259.
2. Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. Relation Classification via Convolutional Deep Neural Network. In Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014), Dublin, Ireland, 23–29 August 2014; pp. 2335–2344.
3. Kambhatla, N. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, Barcelona, Spain, 21–26 July 2004.

4. Bunescu, R.C.; Mooney, R.J. Subsequence Kernels for Relation Extraction. In Proceedings of the Advances in Neural Information Processing Systems 18 (NIPS 2005), Vancouver, BC, Canada, 5–8 December 2005; pp. 171–178.
5. Zelenko, D.; Aone, C.; Richardella, A. Kernel Methods for Relation Extraction. *J. Mach. Learn. Res.* **2003**, *3*, 1083–1106.
6. Zhao, X.; Xiao, C.; Lin, X.; Zhang, W.; Wang, Y. Efficient structure similarity searches: A partition-based approach. *VLDB J.* **2018**, *27*, 53–78. [[CrossRef](#)]
7. Culotta, A.; Sorensen, J.S. Dependency Tree Kernels for Relation Extraction. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, 21–26 July 2004; pp. 423–429.
8. Bunescu, R.C.; Mooney, R.J. A Shortest Path Dependency Kernel for Relation Extraction. In Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), Vancouver, BC, Canada, 6–8 October 2005; pp. 724–731.
9. Zhou, G.; Zhang, M.; Ji, D.; Zhu, Q. Tree Kernel-Based Relation Extraction with Context-Sensitive Structured Parse Tree Information. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007), Prague, Czech Republic, 28–30 June 2007; pp. 728–736.
10. Ren, F.; Li, Y.; Zhao, R.; Zhou, D.; Liu, Z. BiTCNN: A Bi-Channel Tree Convolution Based Neural Network Model for Relation Classification. In Proceedings of the Natural Language Processing and Chinese Computing-7th CCF International Conference (NLPCC 2018), Hohhot, China, 26–30 August 2018; pp. 158–170.
11. Suárez-Paniagua, V.; Segura-Bedmar, I.; Aizawa, A. UC3M-NII Team at SemEval-2018 Task 7: Semantic Relation Classification in Scientific Papers via Convolutional Neural Network. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 793–797.
12. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
13. Dos Santos, C.N.; Xiang, B.; Zhou, B. 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 of the Asian Federation of Natural Language Processing, Beijing, China, 26–31 July 2015; Volume 1, pp. 626–634.
14. Xu, K.; Feng, Y.; Huang, S.; Zhao, D. Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015), Lisbon, Portugal, 17–21 September 2015; pp. 536–540.
15. Nguyen, T.H.; Grishman, R. Relation Extraction: Perspective from Convolutional Neural Networks. In Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, VS@NAACL-HLT 2015, Denver, CO, USA, 31 May–5 June 2015; pp. 39–48.
16. Shen, Y.; Huang, X. Attention-Based Convolutional Neural Network for Semantic Relation Extraction. In Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016), Osaka, Japan, 11–16 December 2016; pp. 2526–2536.
17. 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 (EMNLP 2015), Lisbon, Portugal, 17–21 September 2015; pp. 1785–1794.
18. Nguyen, T.H.; Grishman, R. Combining Neural Networks and Log-linear Models to Improve Relation Extraction. *arXiv* **2015**, arXiv:1511.05926.
19. 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; Volume 2.
20. Miwa, M.; Bansal, M. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; Volume 1.

21. Xiao, M.; Liu, C. Semantic Relation Classification via Hierarchical Recurrent Neural Network with Attention. In Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016), Osaka, Japan, 11–16 December 2016; pp. 1254–1263.
22. Li, F.; Zhang, M.; Fu, G.; Qian, T.; Ji, D. A Bi-LSTM-RNN Model for Relation Classification Using Low-Cost Sequence Features. *arXiv* **2016**, arXiv:1608.07720.
23. Hashimoto, K.; Miwa, M.; Tsuruoka, Y.; Chikayama, T. Simple Customization of Recursive Neural Networks for Semantic Relation Classification. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013), Seattle, WA, USA, 18–21 October 2013; pp. 1372–1376.
24. Yu, M.; Gormley, M.; Dredze, M. Factor-based compositional embedding models. In Proceedings of the NIPS Workshop on Learning Semantics, Montreal, QC, Canada, 8–13 December 2014; pp. 95–101.
25. 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 (EMNLP-CoNLL 2012), Jeju Island, Korea, 12–14 July 2012; pp. 1201–1211.
26. Liu, Y.; Li, S.; Wei, F.; Ji, H. Relation Classification Via Modeling Augmented Dependency Paths. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2016**, *24*, 1589–1598. [[CrossRef](#)]
27. Wang, L.; Cao, Z.; de Melo, G.; Liu, Z. Relation Classification via Multi-Level Attention CNNs. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; Volume 1.
28. Zeiler, M.D. ADADELTA: An Adaptive Learning Rate Method. *arXiv* **2012**, arXiv:1212.5701.

© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).