Improving Cross-Domain Performance for Relation Extraction via Dependency Prediction and Information Flow Control

Amir Pouran Ben Veyseh, Thien Huu Nguyen and Dejing Dou
Department of Computer and Information Science, University of Oregon, OR, USA
{apouranb, thien, dou}@cs.uoregon.edu

Abstract

Relation Extraction (RE) is one of the fundamental tasks in Information Extraction and Natural Language Processing. Dependency trees have been shown to be a very useful source of information for this task. The current deep learning models for relation extraction has mainly exploited this dependency information by guiding their computation along the structures of the dependency trees. One potential problem with this approach is it might prevent the models from capturing important context information beyond syntactic structures and cause the poor cross-domain generalization. This paper introduces a novel method to use dependency trees in RE for deep learning models that jointly predicts dependency and semantics relations. We also propose a new mechanism to control the information flow in the model based on the input entity mentions. Our extensive experiments on benchmark datasets show that the proposed model outperforms the existing methods for RE significantly.

1 Introduction

Extracting semantic relations between entity pairs in text (i.e., Relation Extraction (RE)) is an important task of information extraction. In this paper, we focus on the usual single-sentence setting where two entity mentions appear in the same sentence and the goal is to identify their semantic relationship within this sentence. RE has a wide range of downstream applications, including question answering and knowledge base population.

Among many different approaches, deep learning has proven itself as a very effective method for RE in recent research [Xu *et al.*, 2015a; Nguyen and Grishman, 2016; Wang *et al.*, 2016a; Fu *et al.*, 2017; Shi *et al.*, 2018]. The major factors that contribute to the success of deep learning for RE involve pre-trained word embeddings to generalize words and deep learning architectures to compose the word embeddings to induce effective representations. Recently, dependency trees have also been shown to be useful for deep learning models applied to RE [Miwa and Bansal, 2016; Nguyen and Grishman, 2016; Zhang *et al.*, 2018]. The typical way to exploit dependency trees for RE in deep learning models is to rely on the dependency structures to guide the computations of the models. In particular, the shortest dependency paths between the two entity mentions have been exploited to form sequences of words for Long Short-Term Memory (LSTM) networks [Xu *et al.*, 2015b] while the dependency trees themselves are employed to direct the operation of the graph-based convolutions in recent deep learning models (i.e., Graph Convolutional Neural Networks [Zhang *et al.*, 2018])

Despite the good performance of these methods to exploit dependency information, there are at least two issues that might prevent them from further improving the performance. First, as the information flow in the models is restricted to the structures of the trees, the direct application of the dependency trees in the models might fail to capture important context information that goes beyond the coverage of such tree structures. Second, in the cross-domain setting where the sentences in the training data and test data come from different domains, the dependency structures in the training data might also be dissimilar to those in test data. If a model is trained with the structure guidance for the training data, it might not be able to generalize to the structure in the test data, causing the poor performance in the cross-domain setting.

In order to overcome the aforementioned issues, we introduce a novel method to exploit dependency trees for RE in which the dependency structures of the sentences serve as the ground-truth in a multi-task learning setting to perform both RE and dependency relation prediction simultaneously. In this way, instead of using the dependency trees directly as in the previous approaches, we are using the dependency structures indirectly to encourage the induced representations to be more general with respect to both semantic and dependency relations. Specifically, we first use a modified version of self-attention [Vaswani *et al.*, 2017] to learn a context-aware representation for each token in the sentence. The self-attention representation for each word would encode rich semantic structures, reflecting the semantic portion that one word would contribute to the others in the sentences. Note that such pairwise semantic contributions are crucial to the semantic prediction in RE. Given the representations of words from the self-attention module, we jointly predict the dependency relations between every pair of words in the sentences, causing the word representations to encapsulate both the semantic and syntactic structures of the sentences. Tech-

nically, if we build a semantic adjacency matrix for the words in the sentences based on the pairwise similarities with the self-attention representations of the words, the dependency relation predictions would regulate this semantic adjacency matrix to be similar to the syntactic adjacency matrix induced by the dependency trees. On the one hand, as the semantic self-attention representations of the words have the context information of the whole sentences, once used to make RE prediction, it might help to capture the important context information that the dependency structures cannot reach. On the other hand, the breakdown of the dependency trees into dependency edges/relations in the prediction framework eliminates the emphasis on the whole tree structures that are specific to domains, and focuses on the dependency relations that are shared across domains. This might help to improve the cross-domain performance of the models. Finally, in order to customize the word representations for RE, we propose a novel control mechanism to remove the information that are irrelevant to the semantic prediction of the two entity mentions of interest for RE. In particular, the representations of the two entity mentions are used to compute a semantic control vector that is then applied to the representations of the individual words as a way to retain the most RE-specific information.

Our experiments on the ACE 2005 dataset shows that our model is able to achieve the state-of-the-art performance in the standard datasets for RE. To summarize, our contributions include:

- We introduce a novel method to exploit dependency trees for RE with deep learning based on the predictions of dependency relations.
- We present a novel control mechanism over the feature representations of each word in the sentences to customize the representations for RE.
- We conduct extensive experiments on benchmark datasets and analyze the performance of the model in cross-domain relation extraction.

2 Model

The RE problem in this work is defined as follows: given an input sentence $X = x_1, x_2, \ldots, x_n$ (x_i is the i-th word in the sentence) and the two indexes s and o for the two entity mentions of interest (called relation mention), we would like to predict the semantic relationship between these two mentions. If there is no relation between x_s and x_o, we assign label $None$ for the relation mention. There are three major components in the model proposed in work: (1) Representation Learning: to learn a feature representation for each word based on the semantic and dependency structures of the sentences, (2) Representation Controlling: to determine which features for each token should be used in the final representation based on the two entity mention of interest, and finally (3) to predict the semantic relation for two entity mentions based on the learned representations of the tokens. In the following we describe each part in detail.

2.1 Representation Learning

In order to prepare the input sentence for the neural computation in the following steps, we first transform each word in X into a real-valued representation vector. Inspired by the previous work on relation extraction [Nguyen and Grishman, 2016; Fu *et al.*, 2017], we use vector $w_i = [e_i, ps_i, po_i, t_i, c_i, p_i, g_i]$ to present each word $x_i \in X$ where:

- e_i is some pre-trained word embedding x_i.
- ps_i and po_i are position embedding vectors to indicate the relative distances from the current token x_i to the two entity mentions of interest x_s and x_o (i.e., $i - s$ and $i - o$) respectively.
- t_i and c_i are embedding vectors for the tags of x_i to reflect the entity mention and chunk information in X (following the BIO tagging schema) respectively.
- p_i is a binary number that is 1 if x_i is on the dependency path between x_s and x_o in the dependency tree of X; otherwise it is zero.
- g_i is a binary vector whose size is the total number of dependency relations in dependency trees. The dimension that corresponds to the dependency relation r is set to 1 if x_i has the relation r with some other word in X; and 0 otherwise.

Self-Attention Representation

This word-vector transformation converts the input sentence X into a sequence of representation vectors $W = w_1, w_2, \ldots, w_n$ for the words in X. In this vector sequence, w_i only encapsulates information about the token x_i itself. In order to encode richer context information of the whole sentence into the representations for each word in X, we run a bidirectional LSTM network over W, generating the sequence of hidden vectors $h_1, h_2, \ldots, h_n$. Each hidden vector h_i is the concatenation of the corresponding hidden vectors from the forward and backward LSTM networks that compresses the whole information content of X with a greater focus on x_i. However, for RE, the context information of x_s tends to be less pronounced in h_o (and vice verse) if x_o is far away from x_s in the sentence due to the gated and recurrent mechanisms with the forget gate of LSTM. This is undesirable as the context information of x_s (or x_o) might provide important context information for h_o (or h_s) when it comes to predict the semantic relation between x_s and x_o. For example, the context information of x_s might help to reveal its entity subtype that once integrated well into h_o, can promote h_o as a rich features for the semantic prediction. In order to overcome this issue, we employ the self-attention mechanism that allows a word to directly contribute its context information into the hidden vector of another word only based on the potential semantic contribution, ignoring the distance barriers between the words. Specifically, in the self-attention mechanism, we compute three new vectors k_i (key vector), q_i (query vector) and v_i (value vector) for each token x_i from its hidden vector h_i:

$$\begin{aligned} k_i &= W_k * h_i \\ q_i &= W_q * h_i \\ v_i &= W_v * h_i \end{aligned} \tag{1}$$

where $*$ is the matrix multiplication operation. Note that for simplicity, we omit the biases in the formula for this paper. Afterward, the potential context contribution of x_j for h_i is determined via the similarity between the the key vector k_j of x_j and the query vector of x_i (i.e., the dot product):

$$w_{ij} = \frac{\exp(q_i \cdot k_j)}{\sum_{t=1}^{n} \exp(q_i \cdot k_t)} \quad (2)$$

Given these contribution weights, the self-attention representation vectors for the words in the sentence X is generated by the weighted sums of the value vectors:

$$h'_i = \Sigma_{j=1}^{n} w_{ij} v_j \quad (3)$$

Dependency Relation Prediction

The self-attention mechanism helps the representation vectors h'_i to capture the semantic features of the input sentence X. This section aims to enrich the vectors h'_i with the syntactic structure of X (i.e., the dependency tree) that has been shown to be helpful for deep learning models for RE. As mentioned in the introduction, the traditional methods to use dependency trees to guide the computation of deep learning models would limit the context coverage of the models and cause the poor generalization over dependency structures across domains. In order to avoid such issues, instead of using the dependency trees directly, we break the dependency trees into dependency relations between words that are then employed as the ground-truths to be predicted by the model in a multi-task learning framework for RE. The structure decomposition would help to circumvent the modeling of the whole tree structures to improve the cross-domain generalization while still injecting the syntactic information into the representation vectors via the dependency prediction. In particular, given two words x_i and x_j in the sentence, we first compute the probability $\hat{a}_{i,j}$ to indicate whether x_i and x_j are connected to each other in the dependency tree based on their self-attention representation vectors h'_i and h'_j:

$$\hat{a}_{i,j} = \text{sigmoid}(W_{d2} * g(W_{d1} * [h'_i, h'_j])) \quad (4)$$

where $[u, v]$ is the concatenation operation for the two vectors u and v, W_{d1} and W_{d2} are the model parameters and g is a non-linear function. These probabilities are then employed in a loss function to maximize the likelihood of the dependency connections in the dependency tree:

$$\begin{aligned} L_{dep-pred} = \frac{1}{T} \Sigma_{t=1}^{T} \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} a_{i,j} \log(\hat{a}_{i,j}) \\ + (1 - a_{i,j}) \log(1 - \hat{a}_{i,j}) \end{aligned} \quad (5)$$

where $a_{i,j} = 1$ if there is an edge between tokens x_i and x_j in the dependency tree of X; and 0 otherwise, and T is the number of examples in the training data. Note that our method of predicting dependency relations to enrich word representations for RE is similar to the method employed by [Strubell *et al.*, 2018] for another task of semantic role labeling. However, in the proposed method we predict the existence of the dependency edges between every pair of words in the sentence while [Strubell *et al.*, 2018] only predict the dependency heads of the words in the sentence, just only considering the connected pairs of words in the dependency trees and ignoring the other word pairs of the sentence. In the experiments we found that considering the dependency relations for every pair of words is also important for RE.

2.2 Control Mechanism

In addition to the indirect use of dependency trees, we introduce a new control mechanism for RE that regulates the information flow in the model based on the entity mentions of interest. The rationale for this control mechanism is twofold: (1) for RE, the two entity mentions x_s and s_o are crucial and the effective word representations for this task should retain only the relevant information/features with respect to these two entity mentions. The control mechanism functions as the relevant information filter for RE in this work, and (2) in the attention mechanism we compute a single weight for each word, thus assuming the same weights for every dimension/feature in the word's representation vector. In practice, it might be more flexible if we can regulate the individual dimension/feature so the important dimension/feature for RE would be promoted in the representation vectors. The control mechanism would help to quantify the contribution of each dimension/feature to achieve such flexibility.

The model description so far has introduced two types of representation vectors for the words in the sentence, i.e., the initial contextualized word vectors $H = h_1, h_2, \ldots, h_n$ (i.e., the outputs of the bidirectional LSTM layers) and the semantically and syntactically enriched vectors $H' = h'_1, h'_2, \ldots, h'_n$. With the idea of the control mechanism, we seek to manipulate the word representations in both H and H' at the feature level so the resulting representation vectors would be specific to the two entity mentions x_s and x_o. In particular, we start with the initial contextualized word vectors in H where the hidden vectors h_s and h_o for x_s and x_o are used to generate the control vector p for H via:

$$p = Relu(W_p * [h_s, h_o]) \quad (6)$$

Given the control vector p, we filter the irrelevant information (with respect to x_s and x_o) in the representation vectors of H via the element-wise multiplication $\odot$, transforming each vector $h_i \in H$ into the filtered vector $\bar{h}_i$:

$$\bar{h}_i = p \odot h_i \quad (7)$$

Note that the element-wise multiplication operation allows us to control the representation vectors at the feature level. In the next step, we compute the control vector c for the vectors in H' based on two sources of information specific to x_s and x_o: (i) the initial contextualized vectors h_s and h_o for x_s and x_o (as does for the vectors in H), and (ii) the weighted sum of the vectors in H. In order to generate the weight for each vector in H, we rely on the filtered vectors $\bar{h}_i$ to ensure that the weights are customized for two entity mention of interest:

$$\begin{aligned} \alpha_i = \frac{\exp(W_\alpha \bar{h}_i)}{\sum_{j=1}^{n} \exp(W_\alpha \bar{h}_j)} \\ m = \sum_{i=1}^{n} \alpha_i h_i,\ c = Relu(W_c[m, h_s, h_o]) \end{aligned} \quad (8)$$

The control vector c is then applied to each self-attention vector in $h'_i \in H'$ to produce the final representation vector $\bar{h}'_i$ (via the element-wise multiplication $\odot$) that is both specialized for the two entity mentions, and semantically and syntactically enriched for RE:

$$\bar{h}'_i = c \odot h'_i \quad (9)$$

2.3 Prediction

In the prediction step, we utilize the induced representation vectors in the previous steps to perform the relation prediction for x_s and x_o in X. In particular, following [Zhang *et al.*, 2018], we use the following aggregation vector o as the features for the final prediction:

$$o = [h_s, h_o, \bar{h}'_s, \bar{h}'_o, \max(\bar{h}'_1, \bar{h}'_2, \ldots, \bar{h}'_n)] \quad (10)$$

where $\max(\bar{h}'_1, \bar{h}'_2, \ldots, \bar{h}'_n)$ is the element-wise max operation that retains the highest values along the dimensions of the vectors. Note that the vectors in o capture the context information for the x_s and x_o in X at different levels of abstraction to improve the representatives of the features for RE. In particular, h_s and h_o encode the initial contextualized representation at the basic level while $\bar{h}'_s, \bar{h}'_o$ involve a deeper abstraction level with semantic, syntactic and customized features at the two entity mentions. $\max(\bar{h}'_1, \bar{h}'_2, \ldots, \bar{h}'_n)$ goes one step further to select the most important rich features across the whole sentence. For prediction, the feature vector o would be fed into a two-layer feed forward neural network followed by a softmax layer in the end to compute the probability distribution P_y over the possible relation labels for RE:

$$P(.|X, s, o) = \text{softmax}(W_2 * (W_1 * o)) \quad (11)$$

We employ the negative log-likelihood as the loss function for the relation prediction in this work:

$$L_{label} = -\log P(y|X, s, o) \quad (12)$$

where y is the correct relation label for x_s and x_o in X. Overall, we optimize the following combined loss function L for the model parameters:

$$L = L_{label} + \lambda L_{dep-pred} \quad (13)$$

where λ is the trade-off parameter between the losses for relation prediction and dependency relation prediction we discussed above.

Finally, in order to update the parameters, we use the Adam optimizer with shuffled mini-batches and back-propagation to compute the gradients.

3 Experiments

3.1 Dataset and Parameters

We evaluate the models in this work using two benchmark datasets for RE, i.e., the ACE 2005 dataset [Yu *et al.*, 2015] and the SemEval 2010 Task 8 dataset [Hendrickx *et al.*, 2010].

For the ACE 2005 dataset, following the previous work [Nguyen and Grishman, 2016; Fu *et al.*, 2017; Shi *et al.*, 2018], we use the dataset preprocessed and provided by [Yu *et al.*, 2015] for compatible comparison. The ACE 2005 dataset has 6 different domains: broadcast conversation (`bc`), broadcast news (`bn`), conversational telephone conversation (`cts`), newswire (`nw`), usenet (`un`), and webblogs (`wl`). Similar to the prior work, we use the union of the domains `bn` and `nw` (called `news`) as the training data (with 43497 examples) (called the source domain), a half of the documents in `bc` as the development data, and the remainder (`cts`, `wl` and the other half of `bc`) as the test data (called the target domains). Note that we also the entity mentions, chunks, and dependency trees provided by [Yu *et al.*, 2015] as in the previous work to generate the input features for the words in the sentences. An advantage of the ACE 2005 dataset is it helps to evaluate the cross-domain generalization of the models as the training data and test data in this case comes from different domains.

System	bc	cts	wl	Avg.
FCM [2015]	61.90	52.93	50.36	55.06
Hybrid FCM [2015]	63.48	56.12	55.17	58.25
LRFCM [2015]	59.40	-	-	-
Log-linear [2016]	57.83	53.14	53.06	54.67
CNN [2016]	63.26	55.63	53.91	57.60
Bi-GRU [2016]	63.07	56.47	53.65	57.73
Forward GRU [2016]	61.44	54.93	55.10	57.15
Backward GRU [2016]	60.82	56.03	51.78	56.21
CNN+DANN [2017]	65.16	-	-	-
GSN [2018]	66.38	57.92	56.84	60.38
DRPC	**67.30**	**64.28**	**60.19**	**63.92**

Table 1: F1 scores of the models on the ACE 2005 dataset over different target domains `bc`, `cts`, and `wl`.

For the SemEval 2010 Task 8 dataset [Hendrickx *et al.*, 2010], it comes with 9 directed relations with a special class of *Other*, leading to a 19-class classification problem. In total, there are 8000 training examples in SemEval dataset. As SemEval 2010 does not provide validation data, we use the same model parameters as those used for the ACE 2005 dataset to make it more consistent. We use the official evaluation script for this dataset to obtain the performance of the models as in the prior work [Nguyen and Grishman, 2016; Miwa and Bansal, 2016].

We fine tune the model parameters on the validation data of the ACE 2005 dataset. The parameters we found include: 50 dimensions for position embedding vectors, the entity mention tag vectors and the chunk tag embedding vectors; 100 hidden units for the bidirectional LSTM network in the representation learning component; 200 dimensions for all the hidden vectors in the model (except for the cases to compute attention weights and the input for the last softmax layer to perform prediction); and 0.3 for the learning rate; 0.01 for the trade-off λ in the overall loss function. Finally, we use the pre-trained word embedding `word2vec` to initialize the models.

3.2 Experiments on the ACE 2005 Dataset

Table 1 compares the proposed model (called *DRPC* – Dependency Relation Prediction and Control) with the best reported models on the ACE 2005 dataset in the cross-domain setting for RE. Such best reported models include the Factor-based Compositional Embedding Models (*FCM*) in [Yu *et al.*, 2015], the deep learning models (i.e., *CNN*, *Bi-GRU*) in [Nguyen and Grishman, 2016], the domain adversarial neural network (i.e., *CNN+DANN*) in [Fu *et al.*, 2017] and the current best model with the genre separation network (*GSN*) in [Shi *et al.*, 2018].

As we can see from the table, the proposed model *DRPC* is significantly better than all the previous models on the cross-domain setting for ACE 2005 over different target domains

System	bc	cts	wl	Avg.
CNN [2016]	46.3	40.8	35.8	40.9
GRU [2016]	45.2	40.2	35.1	40.1
Bi-GRU [2016]	46.7	41.2	36.5	41.4
GSN [2018]	52.8	45.3	39.4	45.8
DRPC	**59.81**	**57.82**	**51.24**	**56.29**

Table 2: Performance on the ACE 2005 test sets when linguistic features are not used.

bc, cts and wl ($p < 0.05$). This is remarkable as *DRPC* does not apply any specific techniques to bridge the gap between domains while the previous work relies on such techniques to be able to perform well across domains (i.e., [Fu *et al.*, 2017] and [Shi *et al.*, 2018] with the domain adversarial training). Such performance improvement of *DRPC* demonstrates the effectiveness of the proposed model with self-attention, dependency connection prediction and information flow control in this work.

In order to further evaluate the models, Table 2 reports the performance of the models when the linguistic features for the input vectors in Section 2.1 are not included. In particular, we do not use the embedding vectors t_i and c_i for the entity mention and chunk information, and the p_i and g_i features for the dependency trees in this experiment (i.e., only the word embeddings and the position embeddings are kept). It is clear from Tables 1 and 2 that the performance of the models drops significantly when the linguistic features are excluded. However, the performance of the proposed model *DRPC* still significantly outperform the compared models with large performance gap (an absolute F1 improvement of 7.9%, 18.9% and 17.0% over the state-of-the-art model *GSN* [Shi *et al.*, 2018] for the domains *bc*, *cts* and *wl* respectively).This helps to further testify to the effectiveness of *DRPC* for RE.

3.3 Comparing to Dependency-based Models

The previous section has compared *DRPC* with the state-of-the-art models on the ACE 2005 dataset. This section focuses on the comparison of *DRPC* with the state-of-the-art deep learning models for RE that employ dependency trees in their operation. We perform such comparisons on both the ACE 2005 and SemEval 2018 datasets.

For RE, the best deep learning model with dependency trees is currently the graph convolutional neural network model (i.e., *C-GCN*) in [Zhang *et al.*, 2018] where the dependency trees are used to guide the convolutional operations over the input sentences. We use the implementation of *C-GCN* provided by [Zhang *et al.*, 2018] and evaluate its performance on the ACE 2005 dataset with the cross-domain setting. In addition, we implement the Linguistically-Informed Self-Attention model (*LISA*) in [Strubell *et al.*, 2018] and adapt it to our RE problem for evaluation purpose. Note that although *LISA* is originally designed for semantic role labeling, not for RE, it represents a recently proposed method to exploit dependency trees in deep learning models to predict relations between two words in a sentence with good performance, thus being eligible for a baseline for our work. *C-GCN* and *LISA* only involve the use of dependency trees and do not include the control mechanism as we do in this work. For a fairer comparison, we integrate the control mechanism into such models (leading to *C-GCN* + Control, and *LISA* + Control) and compare them with the proposed model *DRPC* in this work. Table 3 presents the performance of the models on the ACE 2005 dataset.

System	bc	cts	wl	Avg.
C-GCN [2018]	62.55	62.98	55.91	59.24
LISA [2018]	65.04	63.21	55.18	60.13
C-GCN + control	65.68	63.91	58.94	62.29
LISA + control	66.36	63.39	59.17	62.78
DRPC	**67.30**	**64.28**	**60.19**	**63.92**

Table 3: Model's performance on the ACE 2005 test datasets.

System	F1
SVM [2010]	82.2
SDP-LSTM [2015a]	83.7
SPTree [2016]	84.4
PA-Tree [2017]	82.7
C-GCN [2018]	84.8
LISA [2018]	83.9
DRPC	**85.2**

Table 4: Performance on the SemEval 2010 dataset.

The results from the table show that the control mechanism is very useful for RE as it helps to improve the performance for both *C-GCN* and *LISA* over all the three target domains. The improvements are significant except for *LISA* on the cts domain. More importantly, we see that *DRPC* is significantly better than all the compared models over all the target domains with $p < 0.05$, clearly proving the benefits of the dependency relation prediction proposed in this work.

Finally, Table 4 compares *DRPC* with *C-GCN*, *LISA*, and the previous dependency-based methods for RE on the SemEval 2010 dataset. We select the dependency-based models reported in [Zhang *et al.*, 2018] as the baselines, including SVM [Hendrickx *et al.*, 2010], *SDP-LSTM* [Xu *et al.*, 2015a], *SPTree* [Miwa and Bansal, 2016], and *PA-Tree* [Zhang *et al.*, 2017]. The table confirms the effectiveness of *DRPC* that significantly outperforms all the compared methods.

3.4 Ablation Study

Three important components in the proposed model *DRPC* include the self-attention layer (called *SA*), the dependency relation prediction technique (called *DP*), and the control mechanism (called *CM*). In order to evaluate the contribution of these components into the model performance, Table 5 presents the performance of *DRPC* on the ACE 2005 development dataset when such components are excluded one by one from the model. From the table, we can conclude that all the three components *SA*, *DP* and *CM* are important for *DRPC* as removing any of them would further decrease the performance of the model.

3.5 Analysis & Discussion

In this section, we first evaluate the sample complexity of the models to better understand their operation. In particular, we choose different subsets of the ACE 2005 training dataset according to different ratios of the size (i.e., 10%, 20%, 30%

System	Precision	Recall	F1
DRPC	72.10	63.49	67.52
- CM	74.92	60.15	66.88
- DP - CM	71.05	62.00	64.72
- SA - DP - CM	69.02	57.14	61.96

Table 5: Ablation Study. Model's performance on the ACE 2005 developmet set.

	CNN	RNN	C-GCN	LISA	DRPC
100 %	61.95	62.08	64.28	66.72	67.52
90 %	61.75	61.83	62.39	65.48	66.01
80 %	56.91	57.92	58.04	61.74	63.92
70 %	52.87	51.05	52.74	56.81	56.98
60 %	51.35	48.32	45.91	49.92	60.39
50 %	51.34	40.55	43.39	42.84	57.00
40 %	44.53	41.73	36.17	41.50	56.93
30 %	31.45	32.52	28.14	33.86	50.49
20 %	21.13	22.48	24.66	26.75	41.59
10 %	20.85	19.03	19.08	21.91	26.69

Table 6: Complexity analysis of the models. The first columns show how much of training data has been used for training. Performance is on the ACE 2005 development set.

System	bc	cts	wl
CNN	0.70	0.67	0.66
Bi-GRU	0.69	0.66	0.64
C-GCN	0.73	0.76	0.72
LISA	0.71	0.76	0.73
DRPC	**0.75**	**0.78**	**0.74**

Table 7: Average cosine similarity between the representations of the relation mentions in the training and test dataset.

etc.). Such subsets are then used to train the models to evaluate their performance. Table 6 shows the performance of *DRPC* and 4 other baselines, including *CNN* and *Bi-GRU* in [Nguyen and Grishman, 2016], *C-GCN* [Zhang *et al.*, 2018] and *LISA* [Strubell *et al.*, 2018]. As we can see from the table, *DRPC* is significantly better than all the baselines for different amounts of training data, thus demonstrating the better sample complexity of the proposed model for RE.

One of the properties we observe in Tables 1, 2, 3 is that the performance of *DRPC* and *C-GCN* on the `cts` domain is better than those performance on the `bc` and `wl` domains. This is in contrast to the other models where the performance on the `bc` domain is the best, followed by those on `cts` and `wl` [Plank and Moschitti, 2013] (except for *LISA* where the performance on `cts` is close to those on `bc`). In order to understand this problem, we run the trained models over the relation mentions in the training and test datasets of ACE 2005 to obtain the final feature representation vectors (e.g., the vectors o in Section 2.3) for the relation mentions. We then compute the average cosine similarity between the pairs of relation mentions where one element comes from the training dataset and the other element belongs to the test dataset. Table 7 shows such average cosine similarities for different models over different target domains (i.e., `bc`, `cts` and `wl`). The first observation is that the similarity for `cts` is the largest in *DRPC*, *C-GCN* and *LISA* while this is not the case for the other models. This helps to explain the good performance on the `cts` domain of *DRPC* and *C-GCN*. Importantly, we also see that the similarities between the target domains and the source domain (i.e., the training data) for *DRPC* are better than those for the other methods (esp. *CNN* and *Bi-GRU*). In other words, *DRPC* is able to bridge the gap between domains to achieve better generalization for cross-domain RE, thus also explaining the better operation of *DRPC* in this work.

4 Related Work

Relation Extraction is one of the main tasks in Information Extraction. Traditional work has mostly used feature engineering with syntactical information for statistical or kernel based classifiers [Zhou *et al.*, 2005; Bunescu and Mooney, 2005; Sun *et al.*, 2011; Chan and Roth, 2010]. Recently, deep learning has been introduced to solve this problem with typical architectures such as CNN, LSTM and the attention mechanism [Zeng *et al.*, 2014; Nguyen and Grishman, 2015a; Zhou *et al.*, 2016; Wang *et al.*, 2016b; Nguyen and Grishman, 2016; Zhang *et al.*, 2017; Nguyen and Nguyen, 2018b]. Using dependency trees in deep learning models has been shown to be effective for RE [Xu *et al.*, 2015a; Liu *et al.*, 2015; Miwa and Bansal, 2016; Zhang *et al.*, 2018]. In this paper, we also use dependency trees to improve RE performance for deep learning; however, we present a novel method to exploit dependency trees where the dependency relations are predicted in a multi-task learning framework for RE. This hasn't been explored in the previous work for RE.

Cross-domain RE is also a well studied topic. Most of the existing work has investigated genre agnostic features for this setting [Plank and Moschitti, 2013; Nguyen and Grishman, 2014; Yu *et al.*, 2015; Gormley *et al.*, 2015; Nguyen *et al.*, 2015b; Nguyen and Grishman, 2016; Fu *et al.*, 2017; Fu *et al.*, 2018; Shi *et al.*, 2018]. Our work employs the cross-domain setting as the main evaluation for RE. We demonstrate that decomposing the dependency structures to predict the dependency relations is an effective method to improve the generalization of the models for RE.

Regarding multi-task learning with dependency prediction, the most related work to ours is [Strubell *et al.*, 2018] which also predicts the dependency structures in a deep learning model for semantic role labeling. In that approach, for each word, its head in the corresponding dependency tree is predicted while in our approach, we predict the existence of a dependency relation between every pair of words in the sentence. The experiments prove that our approach is more effective for RE.

5 Conclusion

In this paper, we introduce a novel model for relation extraction that uses the information in the dependency trees indirectly in a multi-task learning framework. The model jointly predicts dependency relations between words and relations between entity mentions of interest. Moreover, we propose a new control mechanism that regulates the information flow in the model based on the given entity mentions for RE and the gating techniques. The experiments show that the proposed model achieves the state-of-the-art performance for RE on both the general setting and cross-domain setting.

Acknowledgement

This research is partially supported by the NSF grant CNS-1747798 to the IUCRC Center for Big Learning.

References

[Bunescu and Mooney, 2005] Razvan C Bunescu and Raymond J Mooney. A shortest path dependency kernel for relation extraction. In *EMNLP*, 2005.

[Chan and Roth, 2010] Yee S. Chan and Dan Roth. Exploiting background knowledge for relation extraction. In *COLING*, 2010.

[Fu *et al.*, 2017] Lisheng Fu, Thien Huu Nguyen, Bonan Min, and Ralph Grishman. Domain adaptation for relation extraction with domain adversarial neural network. In *IJCNLP*, 2017.

[Fu *et al.*, 2018] Lisheng Fu, Bonan Min, Thien Huu Nguyen, and Ralph Grishman. A case study on learning a unified encoder of relations. In *Proceedings of the 4th Workshop on Noisy User-generated Text (W-NUT) at EMNLP 2018*, 2018.

[Gormley *et al.*, 2015] Matthew R Gormley, Mo Yu, and Mark Dredze. Improved relation extraction with feature-rich compositional embedding models. *EMNLP*, 2015.

[Hendrickx *et al.*, 2010] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of SEW-2009*, 2010.

[Liu *et al.*, 2015] Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. A dependency-based neural network for relation classification. In *ACL*, 2015.

[Miwa and Bansal, 2016] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. *ACL*, 2016.

[Nguyen and Grishman, 2014] Thien Huu Nguyen and Ralph Grishman. Employing word representations and regularization for domain adaptation of relation extraction. In *ACL*, 2014.

[Nguyen and Grishman, 2015a] Thien Huu Nguyen and Ralph Grishman. Relation extraction: Perspective from convolutional neural networks. In *The NAACL Workshop on Vector Space Modeling for NLP (VSM)*, 2015a.

[Nguyen and Grishman, 2016] Thien Huu Nguyen and Ralph Grishman. Combining neural networks and log-linear models to improve relation extraction. *Proceedings of IJCAI Workshop on Deep Learning for Artificial Intelligence*, 2016.

[Nguyen and Nguyen, 2018b] Minh Nguyen and Thien Huu Nguyen. Who is killed by police: Introducing supervised attention for hierarchical lstms. In *Proceedings of COLING*, 2018b.

[Nguyen *et al.*, 2015b] Thien Huu Nguyen, Barbara Plank, and Ralph Grishman. Semantic representations for domain adaptation: A case study on the tree kernel-based method for relation extraction. In *ACL-IJCNLP*, 2015b.

[Plank and Moschitti, 2013] Barbara Plank and Alessandro Moschitti. Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *ACL*, 2013.

[Shi *et al.*, 2018] Ge Shi, Chong Feng, Lifu Huang, Boliang Zhang, Heng Ji, Lejian Liao, and Heyan Huang. Genre separation network with adversarial training for cross-genre relation extraction. In *EMNLP*, 2018.

[Strubell *et al.*, 2018] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In *EMNLP*, 2018.

[Sun *et al.*, 2011] Ang Sun, Ralph Grishman, and Satoshi Sekine. Semi-supervised relation extraction with large-scale word clustering. In *ACL*, 2011.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[Wang *et al.*, 2016a] Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. Relation classification via multi-level attention cnns. In *ACL*, 2016.

[Wang *et al.*, 2016b] Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. Relation classification via multi-level attention cnns. In *EMNLP*, 2016.

[Xu *et al.*, 2015a] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, 2015.

[Xu *et al.*, 2015b] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, 2015.

[Yu *et al.*, 2015] Mo Yu, Matthew R Gormley, and Mark Dredze. Combining word embeddings and feature embeddings for fine-grained relation extraction. In *NAACL-HLT*, 2015.

[Zeng *et al.*, 2014] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *COLING*, 2014.

[Zhang *et al.*, 2017] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. Position-aware attention and supervised data improve slot filling. In *Proceedings of EMNLP*, pages 35–45, 2017.

[Zhang *et al.*, 2018] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. In *EMNLP*, 2018.

[Zhou *et al.*, 2005] Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. Exploring various knowledge in relation extraction. In *ACL*, 2005.

[Zhou *et al.*, 2016] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *ACL*, 2016.