Exploiting the Syntax-Model Consistency for Neural Relation Extraction

Amir Pouran Ben Veyseh[1], **Franck Dernoncourt**[2], **Dejing Dou**[1] **and Thien Huu Nguyen**[1,3]
[1]Department of Computer and Information Science, University of Oregon, Eugene, Oregon, USA
[2]Adobe Research, San Jose, CA, USA
[3]VinAI Research, Hanoi, Vietnam
{apouranb, dou, thien}@cs.uoregon.edu
franck.dernoncourt@adobe.com

Abstract

This paper studies the task of Relation Extraction (RE) that aims to identify the semantic relations between two entity mentions in text. In the deep learning models for RE, it has been beneficial to incorporate the syntactic structures from the dependency trees of the input sentences. In such models, the dependency trees are often used to directly structure the network architectures or to obtain the dependency relations between the word pairs to inject the syntactic information into the models via multi-task learning. The major problems with these approaches are the lack of generalization beyond the syntactic structures in the training data or the failure to capture the syntactic importance of the words for RE. In order to overcome these issues, we propose a novel deep learning model for RE that uses the dependency trees to extract the syntax-based importance scores for the words, serving as a tree representation to introduce syntactic information into the models with greater generalization. In particular, we leverage Ordered-Neuron Long-Short Term Memory Networks (ON-LSTM) to infer the model-based importance scores for RE for every word in the sentences that are then regulated to be consistent with the syntax-based scores to enable syntactic information injection. We perform extensive experiments to demonstrate the effectiveness of the proposed method, leading to the state-of-the-art performance on three RE benchmark datasets.

1 Introduction

One of the fundamental tasks in Information Extraction (IE) is Relation Extraction (RE) where the goal is to find the semantic relationships between two entity mentions in text. Due to its importance, RE has been studied extensively in the literature. The recent studies on RE has focused on deep learning to develop methods to automatically induce sentence representations from data (Zeng et al., 2014; Nguyen and Grishman, 2015a; Verga et al., 2018). A notable insight in these recent studies is that the syntactic trees of the input sentences (i.e., the dependency trees) can provide effective information for the deep learning models, leading to the state-of-the-art performance for RE recently (Xu et al., 2015; Guo et al., 2019; Tran et al., 2019). In particular, the previous deep learning models for RE has mostly exploited the syntactic trees to structure the network architectures according to the word connections presented in the trees (e.g., performing Graph Convolutional Neural Networks (GCN) over the dependency trees (Zhang et al., 2018)). Unfortunately, these models might not be able to generalize well as the tree structures of the training data might significantly differ from those in the test data (i.e., the models are overfit to the syntactic structures in the training data). For instance, in the cross-domain setting for RE, the domains for the training data and test data are dissimilar, often leading to a mismatch between the syntactic structures of the training data and test data. In order to overcome this issue, the overall strategy is to obtain a more general representation of the syntactic trees that can be used to inject the syntactic information into the deep learning models to achieve better generalization for RE.

A general tree representation for RE is presented in (Veyseh et al., 2019) where the dependency trees are broken down into their sets of dependency relations (i.e., the edges) between the words in the sentences (called the edge-based representation). These dependency relations are then used in a multi-task learning framework for RE that simultaneously predicts both the relation between the two entity mentions and the dependency connections between the pairs of words in the input sentences. Although the dependency connections might be less specific to the training data than the whole tree structures,

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8021–8032
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

the major limitation of the edge-based representation is that it only captures the pairwise (local) connections between the words and completely ignores the overall (global) importance of the words in the sentences for the RE problem. In particular, some words in a given sentence might involve more useful information for relation prediction in RE than the other words, and the dependency tree for this sentence can help to better identify those important words and assign higher importance scores for them (e.g., choosing the words along the shortest dependency paths between the two entity mentions). We expect that introducing such importance information for the words in the deep learning models might lead to improved performance for RE. Consequently, in this work, we propose to obtain an importance score for each word in the sentences from the dependency trees (called the syntax-based importance scores). These will serve as the general tree representation to incorporate the syntactic information into the deep learning models for RE.

How can we employ the syntax-based importance scores in the deep learning models for RE? In this work, we first use the representation vectors for the words from the deep learning models to compute another importance score for each word (called the model-based importance scores). These model-based importance scores are expected to quantify the semantic information that a word contributes to successfully predict the relationship between the input entity mentions. Afterward, we propose to inject the syntax-based importance scores into the deep learning models for RE by enforcing that the model-based importance scores are consistent with the syntactic counterparts (i.e., via the KL divergence). The motivation of the consistency enforcement is to promote the importance scores as the bridge through which the syntactic information can be transmitted to enrich the representation vectors in the deep learning models for RE.

In order to implement this idea, we employ the Ordered-Neuron Long Short-Term Memory Networks (ON-LSTM) (Shen et al., 2019) to compute the model-based importance scores for the words in the sentences for RE. ON-LSTM extends the popular Long Short-Term Memory Networks (LSTM) by introducing two additional gates (i.e., the master forget and input gates) in the hidden vector computation. These new gates controls how long each neuron in the hidden vectors should be activated across different time steps (words) in the sentence (i.e., higher-order neurons would be maintained for a longer time). Based on such controlled neurons, the model-based importance score for a word can be determined by the number of active neurons that the word possesses in the operation of ON-LSTM. To our knowledge, this is the first time ON-LSTM is applied for RE in the literature.

One of the issues in the original ON-LSTM is that the master gates and the model-based importance score for each word are only conditioned on the word itself and the left context encoded in the previous hidden state. However, in order to infer the importance for a word in the overall sentence effectively, it is crucial to have a view over the entire sentence (i.e., including the context words on the right). To this end, instead of relying only on the current word, we propose to obtain an overall representation of the sentence that is used as the input to compute the master gates and the importance score for each word in the sentence. This would enrich the model-based importance scores with the context from the entire input sentences, potentially leading to the improved RE performance of the model in this work.

Finally, to further improve the representations learned by the deep learning models for RE, we introduce a new inductive bias to promote the similarity between the representation vectors for the overall sentences and the words along the shortest dependency paths between the two entity mentions. The intuition is that the relation between the two entity mentions of interest in a sentence for RE can be inferred from either the entire sentence or the shortest dependency path between the two entity mentions (due to the demonstrated ability of the shortest dependency path to capture the important context words for RE in the prior work (Bunescu and Mooney, 2005)). We thus expect that the representation vectors for the sentence and the dependency path should be similar (as both capture the semantic relation) and explicitly exploiting such similarity can help the models to induce more effective representations for RE. Our extensive experiments on three benchmark datasets (i.e., ACE 2005, SPOUSE and SciERC) demonstrate the effectiveness of the proposed model for RE, leading to the state-of-the-art performance for these datasets.

2 Related Work

RE has been traditionally solved by the feature-based or kernel-based approaches (Zelenko et al.,

2003; Zhou et al., 2005; Bunescu and Mooney, 2005; Sun et al., 2011; Chan and Roth, 2010; Nguyen and Grishman, 2014; Nguyen et al., 2015c). One of the issues in these approaches is the requirement for extensive feature or kernel engineering effort that hinder the generalization and applicability of the RE models. Recently, deep learning has been applied to address these problems for the traditional RE approaches, producing the state-of-the-art performance for RE. The typical network architectures for RE include the Convolutional Neural Networks (Zeng et al., 2014; Nguyen and Grishman, 2015a; dos Santos et al., 2015; Wang et al., 2016), Recurrent Neural Networks (Nguyen and Grishman, 2016; Zhou et al., 2016; Zhang et al., 2017; Nguyen et al., 2019a), and self-attentions in Transformer (Verga et al., 2018). The syntactic information from the dependency trees has also been shown to be useful for the deep learning models for RE (Tai et al., 2015; Xu et al., 2015; Liu et al., 2015; Miwa and Bansal, 2016; Peng et al., 2017; Zhang et al., 2018; Guo et al., 2019; Tran et al., 2019; Song et al., 2019; Veyseh et al., 2019). However, these methods tend to poorly generalize to new syntactic structures due to the direct reliance on the syntactic trees (e.g., in different domains) or fail to exploit the syntax-based importance of the words for RE due to the sole focus on edges of the dependency trees (Veyseh et al., 2019).

3 Model

The RE problem can be formulated as a multi-class classification problem. Formally, given an input sentence $W = w_1, w_2, \ldots, w_N$ where w_t is the t-th word in the sentence W of length N, and two entity mentions of interest at indexes s and o ($1 \leq s < o \leq N$), our goal is to predict the semantic relation between w_s and w_o in W.

Similar to the previous work on deep learning for RE (Shi et al., 2018; Veyseh et al., 2019), we first transform each word w_t into a representation vector x_t using the concatenation of the three following vectors: (i) the pre-trained word embeddings of w_t, (ii) the position embedding vectors (to encode the relative distances of w_t to the two entity mentions of interest w_s and w_o (i.e., $t - s$ and $t - o$)), and (iii) the entity type embeddings (i.e., the embeddings of the BIO labels for the words to capture the entity mentions present in X). This word-to-vector transformation converts the input sentence W into a sequence of representation vectors $X = x_1, x_2, \ldots, x_N$ to be consumed by the next neural computations of the proposed model.

There are three major components in the RE model in this work, namely (1) the CEON-LSTM component (i.e., context-enriched ON-LSTM) to compute the model-based importance scores of the words w_t, (2) the syntax-model consistency component to enforce the similarity between the syntax-based and model-based importance scores, and (3) the similarity component between the representation vectors of the overall sentence and the shortest dependency path.

3.1 CEON-LSTM

The goal of this component is to obtain a score for each word w_t that indicates the contextual importance of w_t with respect to the relation prediction between w_s and w_o in W. In this section, we first describe the ON-LSTM model to achieve these importance scores (i.e., the model-based scores). A new model (called CEON-LSTM) that integrates the representation of the entire sentence into the cells of ON-LSTM will be presented afterward.

ON-LSTM: Long-short Term Memory Networks (LSTM) (Hochreiter and Schmidhuber, 1997) has been widely used in Natural Language Processing (NLP) due to its natural mechanism to obtain the abstract representations for a sequence of input vectors (Nguyen and Nguyen, 2018b, 2019). Given the input representation vector sequence $X = x_1, x_2, \ldots, x_N$, LSTM produces a sequence of hidden vectors $H = h_1, h_2, \ldots, h_N$ using the following recurrent functions at the time step (word) w_t (assuming the zero vector for h_0):

$$
\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
\hat{c}_t &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \hat{c}_t \\
h_t &= o_t \circ tanh(c_t)
\end{aligned}
\tag{1}
$$

where f_t, i_t and o_t are called the forget, input and output gates (respectively).

In order to compute the importance score for each word w_t, ON-LSTM introduce into the mechanism of LSTM two additional gates, i.e., the master forget gate $\hat{f}_t$ and the master input gate $\hat{i}_t$ (Shen et al., 2019). These gates are computed and inte-

grated into the LSTM cell as follow:

$$\begin{aligned}
\hat{f}_t &= cummax(W_{\hat{f}} x_t + U_{\hat{f}} h_{t-1} + b_{\hat{f}}) \\
\hat{i}_t &= 1 - cummax(W_{\hat{i}} x_t + U_{\hat{i}} h_{t-1} + b_{\hat{i}}) \\
\bar{f}_t &= \hat{f}_t \circ (f_t \hat{i}_t + 1 - \hat{i}_t) \\
\bar{i}_t &= \hat{i}_t \circ (i_t \hat{f}_t + 1 - \hat{f}_t) \\
c_t &= \bar{f}_t \circ c_{t-1} + \bar{i}_t \circ \hat{c}_t
\end{aligned} \tag{2}$$

where $cummax$ is an activation function defined as $cummax(x) = cumsum(softmax(x))$[1].

The forget and input gates in LSTM (i.e., f_t and i_t) are different from the master forget and input gates in ON-LSTM (i.e., $\hat{f}_t$ and $\hat{i}_t$) as the gates in LSTM assume that the neurons/dimensions in their hidden vectors are equally important and that these neurons are active at every step (word) in the sentence. This is in contrast to the master gates in ON-LSTM that impose a hierarchy over the neurons in the hidden vectors and limit the activity of the neurons to only a portion of the words in the sentence (i.e., higher-ranking neurons would be active for more words in the sentence). Such hierarchy and activity limitation are achieved via the function $cumax(x)$ that aggregates the softmax output of the input vector x along the dimensions. The output of $cumax(x)$ can be seen as the expectation of some binary vector of the form $(0, \ldots, 0, 1, \ldots, 1)$ (i.e., involving two consecutive segments: the 0's segment and the 1's segment). At one step, the 1's segments in the gate vectors represents the neurons that are activated at that step. In ON-LSTM, a word w_i is more contextually important than another word w_j if the master gates for w_i have more active neurons than those for w_j. Consequently, in order to compute the importance score for the word w_t, we can rely on the number of active neurons in the master gates that can be estimated by the sum of the weights of the neurons in the master gates in ON-LSTM. Following (Shen et al., 2019), we employ the hidden vectors for the master forget gate in ON-LSTM to compute the importance scores for the words in this work. Specifically, let $\hat{f}_t = \hat{f}_{t1}, \hat{f}_{t2}, \ldots, \hat{f}_{tD}$ be the weights for the neurons/dimensions in $\hat{h}_t$ (i.e., D is the dimension of the gate vectors). The model-based importance score mod_t for the word $w_t \in W$ is then obtained by: $mod_t = 1 - \sum_{i=1..D} \hat{f}_{ti}$. For convenience, we also use $H = h_1, h_2, \ldots, h_N$ to denote the hidden vectors returned from the application of ON-LSTM over the input representation vectors X.

[1] $cumsum(u_1, u_2, \ldots, u_n) = (u'_1, u'_2, \ldots, u'_n)$ where $u'_i = \sum_{j=1..i} u_j$.

Introducing Sentence Context into ON-LSTM

One limitation of the ON-LSTM model is that it only relies on the representation vector of the current word x_t and the hidden vector for the left context (encoded in h_{t-1}) to compute the master gate vectors and the model-based important score for the word w_t as well. However, this score computation mechanism might not be sufficient for RE as the importance score for w_t might also depend on the context information on the right (e.g., the appearance of some word on the right might make w_t less important for the relation prediction between w_s and w_o). Consequently, in this work, we propose to first obtain a representation vector $x'_t = g(x_1, x_2, \ldots, x_N)$ that has the context information about the entire sentence W (i.e., both the left and right context for the current word w_t). Afterward, x'_t will replace the input representation vector x_t in the computation for the master gates and importance score at step t of ON-LSTM (i.e., in the formulas for $\hat{f}_t$ and $\hat{i}_t$ in Equation 2). In this way, the model-based importance score for w_t will be able to condition on the overall context in the input sentence.

In this work, we obtain the representation vector x'_t for each step t of ON-LSTM based on the weighted sum of the transformed vectors of the input representation sequence $x_1, x_2, \ldots, x_N$: $x'_t = \sum_i \alpha_{ti}(W_x x_i + b_x)$. The weight α_{ti} for the term with x_i in this formula is computed by:

$$\alpha_{ti} = \frac{\exp((W_h h_{t-1} + b_h) \cdot (W_x x_i + b_x))}{\sum_{j=1}^{N} \exp((W_h h_{t-1} + b_h) \cdot (W_x x_j + b_x))} \tag{3}$$

where W_h, b_h, W_x and b_x are the learnable parameters. Note that in this formula, we use the ON-LSTM hidden vector h_{t-1} from the previous step as the query vector to compute the attention weight for each word. The rationale is to enrich the attention weights for the current step with the context information from the previous steps (i.e., encoded in h_{t-1}), leading to the contextualized input representation x'_t with richer information for the master gates and importance score computations in ON-LSTM. The proposed ON-LSTM with the enriched input vectors x'_t is called CEON-LSTM (i.e., Context-Enriched ON-LSTM) in this work.

3.2 Syntax-Model Consistency

As mentioned in the introduction, the role of the model-based importance scores obtained from CEON-LSTM is to serve as the bridge to inject the information from the syntactic structures of W into the representation vectors of the deep learning models for RE. In particular, we first leverage the dependency tree of W to obtain another importance score syn_t for each word $w_t \in W$ (i.e., the syntax-based importance score). Similar to the model-based scores, the syntax-based scores are expected to measure the contextual importance of w_t with respect to the relation prediction for w_s and w_o. Afterward, we introduce a constraint to encourage the consistency between the model-based and syntax-based importance scores (i.e., mod_t and syn_t) for the words via minimizing the KL divergence L_{import} between the normalized scores:

$$\begin{aligned} \overline{mod}_1, \ldots, \overline{mod}_N &= softmax(mod_1, \ldots, mod_N) \\ \overline{syn}_1, \ldots, \overline{syn}_N &= softmax(syn_1, \ldots, syn_N) \\ \mathcal{L}_{import} &= -\Sigma_i \overline{mod}_i log \frac{\overline{mod}_i}{\overline{syn}_i} \end{aligned} \tag{4}$$

The intuition is to exploit the consistency to supervise the model-based importance scores from the models with the syntax-based importance scores from the dependency trees. As the model-based importance scores are computed from the master gates with the active and inactive neurons in CEON-LSTM, this supervision allows the syntactic information to interfere directly with the internal computation/structure of the cells in CEON-LSTM, potentially generating representation vectors with better syntax-aware information for RE.

To obtain the syntax-based importance scores, we take the motivation from the previous work on RE where the shortest dependency paths between the two entity mentions of interest have been shown to capture many important context words for RE. Specifically, for the sentence W, we first retrieve the shortest dependency path DP between the two entity mentions w_s and w_o and the length T of the longest path between any pairs of words in the dependency tree of W. The syntax-based importance score syn_t for the word $w_t \in W$ is then computed as the difference between T and the length of the shortest path between w_t and some word in DP in the dependency tree (i.e., the words along DP will have the score of T). On the one hand, these syntax-based importance scores are able to capture the importance of the words that is customized for the relation prediction between w_s and w_o. This is better suited for RE than the direct use of the edges in the dependency trees in (Veyseh et al., 2019) that is agnostic to the entity mentions of interest and fails to encode the importance of the words for RE. On the other hand, the syntax-based importance scores syn_t represent a relaxed form of the original dependency tree that might have a better chance to generalize over different data and domains for RE than the prior work (i.e., the ones that directly fit the models to the whole syntactic structures (Zhang et al., 2018) and run the risk of overfitting to the structures in the training data).

3.3 Sentence-Dependency Path Similarity

In this component, we seek to further improve the representation vectors in the proposed deep learning model for RE by introducing a novel constraint to maximize the similarity between the representation vectors for the overall input sentence W and the words along the shortest dependency path DP (i.e., inductive bias). The rationale for this bias is presented in the introduction.

In order to implement this idea, we first obtain the representation vectors R_W and R_{DP} for the sentence W and the words along DP (respectively) by applying the max-pooling operation over the CEON-LSTM hidden vectors $h_1, h_2, \ldots, h_N$ for the words in W and DP: $R_W = max\_pooling_{w_i \in W}\{h_i\}$ and $R_{DP} = max\_pooling_{w_i \in DP}\{h_i\}$. In the next step, we promote the similarity between R_W and R_{DP} by explicitly minimizing their negative cosine similarity[2], i.e., adding the following term L_{path} into the overall loss function:

$$L_{path} = 1 - \cos\left(R_W, R_{DP}\right) \tag{5}$$

3.4 Prediction

Finally, in the prediction step, following the prior work (Veyseh et al., 2019), we employ the following vector V as the overall representation vector to predict the relation between w_s and w_o in W: $V = [x_s, x_o, h_s, h_o, R_W]$. Note that V involves the information at different abstract levels for W, i.e., the raw input level with x_s and x_o, the abstract representation level with h_s and h_o

[2]We tried the KL divergence and the mean square error for this, but cosine similarity achieved better performance.

from CEON-LSTM, and the overall sentence vector R_W. In our model, V would be fed into a feed-forward neural network with the softmax layer in the end to estimate the probability distribution $P(.|W, w_s, w_o)$ over the possible relations for W. The negative log-likelihood function is then obtained to serve as the loss function for the model: $L_{label} = -\log P(y|W, w_s, w_o)$ (y is the golden relation label for w_s and w_o in W). Eventually, the overall loss function of the model in this work is:

$$\mathcal{L} = \mathcal{L}_{label} + \alpha\mathcal{L}_{import} + \beta\mathcal{L}_{path} \quad (6)$$

where α and β are trade-off parameters. The model is trained with shuffled mini-batching.

4 Experiments

4.1 Datasets and Hyper-parameters

We evaluate the models in this work using three benchmark datasets, i.e., ACE 2005, SPOUSE, and SciERC. For ACE 2005, similar to the previous work (Nguyen and Grishman, 2016; Fu et al., 2017; Shi et al., 2018; Veyseh et al., 2019), we use the dataset preprocessed and provided by (Yu et al., 2015) for compatible comparison. There are 6 different domains in this dataset, i.e., (`bc`, `bn`, `cts`, `nw`, `un`, and `wl`), covering text from news, conversations and web blogs. Following the prior work, the union of the domains `bn` and `nw` (called `news`) is used as the training data (called the source domain); a half of the documents in `bc` is reserved for the development data, and the remainder (`cts`, `wl` and the other half of `bc`) serve as the test data (called the target domains). This data separation facilitates the evaluation of the cross-domain generalization of the models due to the domain difference of the training and test data.

The SPOUSE dataset is recently introduced by (Hancock et al., 2018), involving 22,195 sentences for the training data, 2,796 sentences for the validation data, and 2,697 sentences for the test data. Each sentence in this dataset contains two marked person names (i.e., the entity mentions) and the goal is to identify whether the two people mentioned in the sentence are spouses.

Finally, the SciERC dataset (Luan et al., 2018) annotates 500 scientific abstracts for the entity mentions along with the coreferences and relations between them. For RE, this dataset provides 3,219 sentences in the training data, 455 sentences in the validation data and 974 sentences in the test data.

We fine tune the hyper-parameters for the models in this work on the validation data of the ACE 2005 dataset. The best parameters suggested by this process include: 30 dimensions for the position embeddings and entity type embeddings, 200 hidden units for the CEON-LSTM model and all the other hidden vectors in the model (i.e., the hidden vectors in the final feed-forward neural network (with 2 layers) and the intermediate vectors in the weighted sum vector for x'_t), 1.0 for both loss trade-off parameters α and β, and 0.001 for the initial learning rate with the Adam optimizer. The batch size is set to 50. Finally, we use either the uncontextualized word embeddings `word2vec` (with 300 dimensions) or the hidden vectors in the last layer of the BERT$_{base}$ model (with 768 dimensions) (Devlin et al., 2019) to obtain the pre-trained word embeddings for the sentences (Devlin et al., 2019). We find it better to fix BERT in the experiments. Note that besides this section, we provide some additional analysis for the models in the Appendix.

4.2 Comparison with the state of the art

We fist compare the proposed model (called CEON-LSTM) with the baselines on the popular ACE 2005 dataset. In particular, the four following groups of RE models in the prior work on RE with the ACE 2005 dataset is chosen for comparison:

(i) Feature based models: These models handdesign linguistic features for RE, i.e., FCM, Hybrid FCM, LRFCM, and SVM (Yu et al., 2015; Hendrickx et al., 2010).

(ii) Deep sequence-based models: These models employ deep learning architectures based on the sequential order of the words in the sentences for RE, i.e., log-linear, CNN, Bi-GRU, Forward GRU, Backward GRU (Nguyen and Grishman, 2016), and CNN+DANN (Fu et al., 2017).

(iii) Adversarial learning model: This model, called GSN, attempts to learn the domain-independent features for RE (Shi et al., 2018).

(iv) Deep structure-based models: These models use dependency trees either as the input features or the graphs to structure the network architectures in the deep learning models. The state-of-the-art models of this type include: AGGCN (Attention Guided GCN) (Guo et al., 2019), SACNN (Segment-level Attention-based CNN) (Tran et al., 2019) and DRPC (the Dependency Relation Prediction and Control model) (Veyseh et al., 2019). DRPC has the best reported performance on ACE

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
C-GCN (2018)	65.55	62.98	55.91	61.48
AGGCN (2019)	63.47	59.70	56.50	59.89
SACNN (2019)	65.06	61.71	59.82	62.20
DRPC (2019)	67.30	64.28	60.19	63.92
CEON-LSTM (ours)	**68.55**	**65.42**	**61.93**	**65.30**

Table 1: F1 scores of the models on the ACE 2005 test datasets using the `word2vec` word embeddings.

2005. Note that we obtain the performance of these models on the considered datasets using the actual implementation released by the original papers.

Most of the prior RE work on the ACE 2005 dataset uses the uncontextualized word embeddings (i.e., `word2vec`) for the initial word representation vectors. In order to achieve a fair comparison with the baselines, we first show the performance of the models (i.e., the F1 scores) on the ACE 2005 test datasets when `word2vec` is employed for the pre-trained word embeddings in Table 1. The first observation from the table is that the deep structured-based models (e.g., C-GCN, DRPC) are generally better than the deep sequence-based models (e.g., CNN, Bi-GRU) and the feature base models with large performance gaps. This demonstrates the benefits of the syntactic structures that can provide useful information to improve the performance for the deep learning models for RE. We will thus focus on these deep structure-based models in the following experiments. Among all the models, we see that the proposed model CEON-LSTM is significantly better than all the baseline models over different test domains/datasets. In particular, CEON-LSTM is 1.38% and 3.1% better than DRPC and SACNN (respectively) on the average F1 scores over different test datasets. These performance improvements are significant with $p < 0.01$ and clearly demonstrate the effectiveness of the proposed CEON-LSTM model for RE.

In order to further compare CEON-LSTM with the baselines, Table 2 presents the performance of the models when the words are represented by the contextualized word embeddings (i.e., BERT). For this case, we also report the performance of the recent BERT-based model (i.e., Entity-Aware BERT (EA-BERT)) in (Wang et al., 2019) for RE

System	bc	cts	wl	Avg.
C-GCN (2018)	67.02	64.4	58.92	63.44
AGGCN (2019)	65.29	63.65	60.35	63.09
SACNN (2019)	68.52	64.21	62.19	64.97
DRPC (2019)	69.41	65.82	61.65	65.62
EA-BERT (2019)	69.25	61.70	58.48	63.14
CEON-LSTM (ours)	**71.58**	**66.92**	**65.17**	**67.89**

Table 2: F1 scores of the models on the ACE 2005 test datasets using the BERT word embeddings.

System	SPOUSE	SciERC
C-GCN (`word2vec`) (2018)	73.52	65.30
AGGCN (`word2vec`) (2019)	73.51	67.91
SACNN (`word2vec`) (2019)	72.88	67.54
DRPC (`word2vec`) (2019)	74.66	68.18
CEON-LSTM (`word2vec`) (ours)	76.43	69.92
C-GCN (BERT) (2018)	75.18	74.11
AGGCN (BERT) (2019)	76.91	75.77
SACNN (BERT) (2019)	77.98	76.42
DRPC (BERT) (2019)	78.93	77.21
CEON-LSTM (BERT) (ours)	**81.01**	**78.24**

Table 3: F1 scores of the models on the SPOUSE and SciERC datasets.

on the ACE 2005 dataset. Comparing the models in Table 2 with the counterparts in 1, it is clear that the contextualized word embeddings can significantly improve the deep structure-based models for RE. More importantly, similar to the case with `word2vec`, we see that the proposed model CEON-LSTM still significantly outperforms all the baselines models with large performance gaps and $p < 0.01$, further testifying to the benefits of the CEON-LSTM model in this work.

Finally, in order to demonstrate the generalization of the proposed model over the other datasets, we show the performance of the models on the two other datasets in this work (i.e., SPOUSE and SciERC) using either `word2vec` or BERT as the word embeddings in Table 3. The results clearly confirm the effectiveness of CEON-LSTM as it is significantly better than all the other models over different datasets and word embedding settings.

4.3 Ablation Study

The Effect of the Model Components: There are three major components in the proposed model: (1) the introduction of the overall sentence representation x'_t into the ON-LSTM cells (called **SCG** – Sentence Context for Gates), (2) the consistency constraint for the syntax-based and model-based importance scores (called **SMC** – Syntax-Semantic Consistency), and (3) the similarity constraint for the representation vectors of the overall sentence and the shortest dependency path (called **SDPS** – Sentence-Dependency Path Similarity). In order to

System	P	R	F1
CEON-LSTM (Full)	**74.51**	**67.29**	**71.08**
- SCG	74.00	66.98	70.45
- SMC	72.87	66.85	69.89
- SDPS	73.02	66.00	69.18
- SCG - SMC	71.52	64.62	68.08
- SCG - SDPS	70.33	64.22	67.17
- SMC - SDPS	71.02	63.95	67.58
- SCG - SMC - SDPS	70.51	63.01	66.98

Table 4: Ablation study on the development set of ACE 2005. The components listed in each row are removed from the overall model.

System	P	R	F1
CEON-LSTM (proposed)	**74.51**	**67.29**	**71.08**
Bi-ON-LSTM	72.65	67.17	69.28
SA-ON-LSTM	73.21	67.31	70.13
CE-LSTM	71.58	64.19	67.92
EP-ON-LSTM	71.03	65.16	68.45
SP-CEON-LSTM (R_W in V)	73.58	66.92	70.13
SP-CEON-LSTM (R_W not in V)	72.94	65.21	69.51

Table 5: Models' performance on the development dataset of ACE 2005.

evaluate the contribution of these components for the overall model CEON-LSTM, we incrementally remove these components from CEON-LSTM and evaluate the performance of the remaining model. Table 4 reports the performance of the models on the ACE 2005 development dataset.

It is clear from the table that all the components are necessary for the proposed model as excluding any of them would hurt the performance significantly. It is also evident that removing more components results in more performance drop, thus demonstrating the complementary nature of the three proposed components in this work.

The Variants for CEON-LSTM: We study several variants of **SCG**, **SMC**, and **SDPS** in CEON-LSTM to demonstrate the effectiveness of the designed mechanisms. In particular, we consider the following alternatives for CEON-LSTM:

(i) Bi-ON-LSTM: Instead of employing the attention-based representation vectors x'_t to capture the context of the entire input sentence for the model-based importance scores in **SCG**, we run two unidirectional ON-LSTM models (i.e., the forward and backward ON-LSTM) to compute the forward and backward importance scores for each word in W. The final model-based importance score for each word is then the average of the corresponding forward and backward scores.

(ii) SA-ON-LSTM: In this method, instead of using the hidden vector h_{t-1} as the query vector to compute the attention weight α_{ti} in Equation 3 for **SCG**, we utilize the input representation vector x_t for w_t as the query vector (i.e., replace h_{t-1} with x_t in Equation 3). Consequently, SA-ON-LSTM is basically a composed model where we first run the self-attention (SA) model (Vaswani et al., 2017) over X. The results are then fed into ON-LSTM to obtain the model-based importance scores mod_t.

(iii) CE-LSTM: This aims to explore the effectiveness of ON-LSTM for our model. In CE-LSTM, we replace the ON-LSTM network with the usual LSTM model in CEON-LSTM. The **SMC** component is not included in this case as the LSTM model cannot infer the importance scores.

(iv) EP-ON-LSTM: Before this work, the DRPC model in (Veyseh et al., 2019) has the state-of-the-art on ACE 2005. Both DRPC and CEON-LSTM apply a more general representation of the dependency trees in a deep learning model (i.e., avoid directly using the original trees to improve the generalization). To illustrate the benefit of the importance score representation for **SMC**, EP-ON-LSTM replaces the importance score representation for the dependency trees in CEON-LSTM with the dependency edge representation in DRPC. In particular, we replace the term L_{import} in the overall loss function (i.e., Equation 6) with the dependency edge prediction loss (using the ON-LSTM hidden vectors) in DRPC for EP-ON-LSTM.

(v) SP-CEON-LSTM: This model removes the **SDPS** component and includes the representation vector of the dependency path DP (i.e., R_{DP}) in the final representation V for relation prediction. We consider both retaining and excluding the sentence representation R_W in V in this case. This model seeks to show that the use of R_{DP} for the similarity encouragement with R_W is more effective than employing R_{DP} directly in V.

Table 5 reports the performance of these CEON-LSTM variations on the ACE 2005 development dataset. As we can see from the table, all the considered variants have significantly worse performance than CEON-LSTM (with $p < 0.005$). This clearly helps to justify the designs of the components **SCG**, **SMC** and **SDPS** for CEON-LSTM in this work.

Baseline for the Model-Based Importance Scores: One of the contributions in our work is to employ the gates in the cells of ON-LSTM to obtain the model-based importance scores that are then used to promote the consistency with the syntax-based importance scores (i.e., in the **SMC** compo-

System	P	R	F1
CEON-LSTM (proposed)	**74.51**	**67.29**	**71.08**
HIS-CEON-LSTM	72.02	63.97	68.29

Table 6: Models' performance on the development dataset of ACE 2005.

nent). In order to demonstrate the effectiveness of the master cell gates to obtain the model-based importance scores, we evaluate a typical baseline where the model-based importance score mod_i for $w_i \in W$ is computed directly from the hidden vector h_i of CEON-LSTM (i.e., by feeding h_i into a feed-forward neural network with *sigmoid* activation function in the end). The model-based importance scores obtained in this way then replace the importance scores from the cell gates and are used in the **SMC** component of CEON-LSTM in the usual way (i.e., via the KL divergence in L_{import}) (note that we tried the alternatives for the KL divergence in L_{import} (i.e., the mean square error and the cosine similarity between the syntax-based and model-based importance scores), but the KL divergence produced the best results for both CEON-LSTM and HIS-CEON-LSTM on the development data). The resulting model is called **HIS-CEON-LSTM**. Table 6 reports the performance of HIS-CEON-LSTM and the proposed model CEON-LSTM on the ACE 2005 development dataset. It is clear from this table that the proposed model CEON-LSTM achieves significantly better performance than HIS-CEON-LSTM (with large performance gap), thus testifying to the importance of the master gates to obtain the model-based importance scores for CEON-LSTM.

5 Conclusion

We introduce a new deep learning model for RE (i.e., CEON-LSTM) that features three major proposals. First, we represent the dependency trees via the syntax-based importance scores for the words in the input sentences for RE. Second, we propose to incorporate the overall sentence representation vectors into the cells of ON-LSTM, allowing it to compute the model-based importance scores more effectively. We also devise a novel mechanism to project the syntactic information into the computation of ON-LSTM via promoting the consistency between the syntax-based and model-based importance scores. Finally, we present a novel inductive bias for the deep learning models that exploits the similarity of the representation vectors for the whole input sentences and the shortest dependency paths between the two entity mentions for RE. Extensive experiments are conducted to demonstrate the benefits of the proposed model. We achieve the state-of-the-art performance on three datasets for RE. In the future, we plan to apply CEON-LSTM to other related NLP tasks (e.g., Event Extraction, Semantic Role Labeling) (Nguyen et al., 2016a; Nguyen and Grishman, 2018a).

Acknowledgments

This research has been supported in part by Vingroup Innovation Foundation (VINIF) in project code VINIF.2019.DA18, the NSF grant CNS-1747798 to the IUCRC Center for Big Learning, and a gift from Adobe Research. This research is also based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

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

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

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

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *ACL*.

Braden Hancock, Martin Bringmann, Paroma Varma, Percy Liang, Stephanie Wang, and Christopher Ré. 2018. Training classifiers with natural language explanations. In *ACL*.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *EMNLP*.

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

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

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016a. Joint event extraction via recurrent neural networks. In *NAACL*.

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

Thien Huu Nguyen and Ralph Grishman. 2015a. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st NAACL Workshop on Vector Space Modeling for NLP (VSM)*.

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

Thien Huu Nguyen and Ralph Grishman. 2018a. Graph convolutional networks with argument-aware pooling for event detection. In *AAAI*.

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

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *AAAI*.

Tuan Ngo Nguyen, Franck Dernoncourt, and Thien Huu Nguyen. 2019a. On the effectiveness of the pooling methods for biomedical relation extraction with deep learning. In *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019)*.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *ACL*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *ICLR*.

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

Linfeng Song, Yue Zhang, Daniel Gildea, Mo Yu, Zhiguo Wang, and jinsong su. 2019. Leveraging dependency forest for neural medical relation extraction. In *EMNLP-IJCNLP*.

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

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*.

Van-Hien Tran, Van-Thuy Phi, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Relation classification using segment-level attention-based cnn and dependency-based rnn. In *NAACL-HLT*.

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

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *EMNLP*.

Amir Pouran Ben Veyseh, Thien Huu Nguyen, and Dejing Dou. 2019. Improving cross-domain performance for relation extraction via dependency prediction and information flow control. In *IJCAI*.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019. Extracting multiple-relations in one-pass with pre-trained transformers. In *ACL*.

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

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

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

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research*, 3:1083–1106.

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

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

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *EMNLP*.

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

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

A Analysis

In order to provide more insights into the performance of the proposed model, we analyze examples in the test data that can be predicted correctly with the proposed model and incorrectly with the baselines. For a baseline model M (e.g., GCN, DRPC), we call the test examples that cannot be recognized by M but can be successfully predicted by the proposed model the M-failure examples. Based on our analysis, the GCN-failure examples tend to involve the syntactic/dependency structures that does not appear or are not well represented in the training data. Some examples for the GCN-failure examples are shown in Table 7. On the one hand, as GCN is directly dependent on the syntactic structures of the input sentences, it would not be able to learn effective representations for the sentences with new structures in the GCN-failure examples for RE. On the other hand, as CEON-LSTM only exploits a relaxed general form of the tree structures (i.e., the importance scores of the words), it will be able to generalize better to the new structures in the GCN-failure examples where the general tree form is still helpful to induce effective representations for RE.

For the DRPC-failure examples (their examples are presented in Table 8), we find that these examples often involve the two entity mentions of interest with long distance from each other in the input sentences. For these examples, the dependency paths between the two entity mentions tend to be very helpful or crucial for RE as they can capture the important context words (thus eliminating the irrelevant ones). This allows the models to learn effective representations to correctly predict the relations in the sentences for RE. As DRPC only retains the dependency edges in the dependency trees separately (i.e., the local tree representations), it cannot directly capture such dependency paths, thereby failing to predict the relations for the DRPC-failure examples with long distances between the entities. This is in contrast to CEON-LSTM that exploits the global representations of the trees with the importance scores based on the distances of the words to the dependency paths. As the dependency paths can be still inferred in this global representation, CEON-LSTM can benefit from this information to successfully perform RE for the sentences in the DRPC-failure examples.

Sentence	Relation
Some **Arab countries** also want to play a role in the stability operation in Iraq but are reluctant to send **troops** because of political, religious and ethnic considerations, the official said.	ORG-AFF
Some suggested that **Russian President Vladimir Putin** will now be scrambling to contain the damage to his once -budding friendship with US President **George W. Bush** because he was poorly advised by his intelligence and defense aides.	PER-SOC
Other **countries** including the Philippines, South Korea, Qatar and Australia agreed to send other help such as field **hospitals**, engineers, explosive ordnance disposal teams or nuclear, biological and chemical weapons experts.	PART-WHOLE

Table 7: The GCN-failure examples. The two entity mentions of interest are shown in bold in the sentences.

Sentence	Relation
US diplomats have hinted in recent weeks that Washington 's anger with **European** resistance to the campaign was focused more on Paris –and to a lesser extent Berlin– than it was with **Moscow**.	PART-WHOLE
In **Montreal**, "Stop the War" a coalition of more than 190 groups, said as many as 200,000 **people** turned out, though police refused to give a figure.	PHYS
Although the **crossing** has, in principle, been open for movement between the two territories –while being frequently closed by **Israeli** for reasons rarely explained– the Palestinian section has been manned by Israel for more than two years.	ART

Table 8: The DRPC-failure examples. The two entity mentions of interest are shown in bold in the sentences.