Leveraging Dependency Forest for Neural Medical Relation Extraction

Linfeng Song[1,2], **Yue Zhang**[3,4]*, **Daniel Gildea**[1], **Mo Yu**[5], **Zhiguo Wang**[6] and **Jinsong Su**[7]
[1]University of Rochester, Rochester, NY, USA
[2]Tencent AI Lab, Bellevue, WA, USA
[3]Institute of Advanced Technology, Westlake Institute for Advanced Study
[4]School of Engineering, Westlake Universtiy
[5]IBM T.J. Watson Research, Yorktown Heights, NY, USA
[6]Amazon AWS, New York, NY, USA
[7]Xiamen University, Xiamen, China

Abstract

Medical relation extraction discovers relations between entity mentions in text, such as research articles. For this task, dependency syntax has been recognized as a crucial source of features. Yet in the medical domain, 1-best parse trees suffer from relatively low accuracies, diminishing their usefulness. We investigate a method to alleviate this problem by utilizing dependency forests. Forests contain many possible decisions and therefore have higher recall but more noise compared with 1-best outputs. A graph neural network is used to represent the forests, automatically distinguishing the useful syntactic information from parsing noise. Results on two biomedical benchmarks show that our method outperforms the standard tree-based methods, giving the state-of-the-art results in the literature.

1 Introduction

The sheer amount of medical articles and their rapid growth prevent researchers from receiving comprehensive literature knowledge by direct reading. This can hamper both medical research and clinical diagnosis. NLP techniques have been used for automating the knowledge extraction process from the medical literature (Friedman et al., 2001; Yu and Agichtein, 2003; Hirschman et al., 2005; Xu et al., 2010; Sondhi et al., 2010; Abacha and Zweigenbaum, 2011). Along this line of work, a long-standing task is relation extraction, which mines factual knowledge from free text by labeling relations between entity mentions. As shown in Figure 1, the sub-clause "previously observed cytochrome P450 3A4 (CYP3A4) interaction of the dual orexin receptor antagonist almorexant" contains two entities, namely "orexin receptor" and "almorexant". There is an "adversary" relation between these two entities, denoted as"CPR:6".

*Yue Zhang is the corresponding author

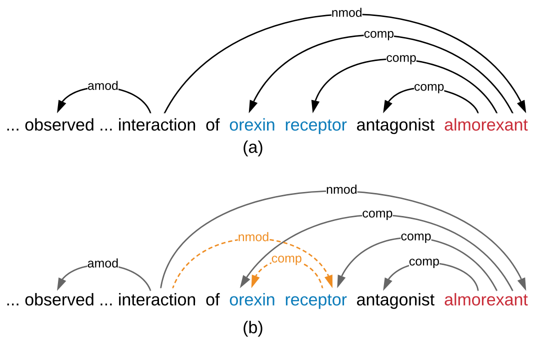

nmod
comp
comp
comp
amod
... observed ... interaction of orexin receptor antagonist almorexant
(a)
nmod
comp
nmod
comp
comp
comp
amod
... observed ... interaction of orexin receptor antagonist almorexant
(b)

Figure 1: (a) 1-best dependency tree and (b) dependency forest for a medical-domain sentence, where edge label "comp" represents "compound". Associated mentions are in different colors. Some irrelevant words and edges are omitted for simplicity.

Previous work has shown that dependency syntax is important for guiding relation extraction (Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Liu et al., 2015; Gormley et al., 2015; Xu et al., 2015a,b; Miwa and Bansal, 2016; Zhang et al., 2018b), especially in biological and medical domains (Quirk and Poon, 2017; Peng et al., 2017; Song et al., 2018b). Compared with sequential surface-level structures, such as POS tags, dependency trees help to model word-to-word relations more easily by drawing direct connections between distant words that are syntactically correlated. Take the phrase "effect on the medicine" for example; "effect" and "medicine" are directly connected in a dependency tree, regardless of how many modifiers are added in between.

Dependency parsing has achieved an accuracy over 96% in the news domain (Liu and Zhang, 2017; Kitaev and Klein, 2018). However, for the medical literature domain, parsing accuracies can drop significantly (Lease and Charniak, 2005; McClosky and Charniak, 2008; Sagae et al., 2008; Candito et al., 2011). This can lead to severe error propagation in downstream relation extraction tasks, offsetting much of the benefit that relation extraction models can obtain by exploiting dependency trees as a source of external features.

We address the low-accuracy issue in biomedical dependency parsing by considering dependency forests as external features. Instead of 1-best trees, dependency forests consist of dependency arcs and labels that a parser is relatively confident about, therefore having better recall of gold-standard arcs by offering more candidate choices with noise. Our main idea is to let a relation extraction system learn automatically from a forest which arcs are the most relevant through end-task training, rather than relying solely on the decisions of a noisy syntactic parser. To this end, a graph neural network is used for encoding a forest, which in turn provides features for relation extraction. Back-propagation passes loss gradients from the relation extraction layer to the graph encoder, so that the more relevant edges can be chosen automatically for better relation extraction.

Results on BioCreative VI ChemProt (CPR) (Krallinger et al., 2017) and a recent dataset focused on phenotype-gene relations (PGR) (Sousa et al., 2019) show that our method outperforms a strong baseline that uses 1-best dependency trees as features, giving the state-of-the-art accuracies in the literature. To our knowledge, we are the first to study dependency forests for medical information extraction, showing their advantages over 1-best tree structures. Our code is available at `http://github.com/freesunshine/dep-forest-re`.

2 Related work

Syntactic forests There have been previous studies leveraging constituent forests for machine translation (Mi et al., 2008; Ma et al., 2018; Zaremoodi and Haffari, 2018), sentiment analysis (Le and Zuidema, 2015) and text generation (Lu and Ng, 2011). However, the usefulness of dependency forests is relatively rarely studied, with one exception being Tu et al. (2010), who use dependency forests to enhance long-range word-to-word dependencies for statistical machine translation. To our knowledge, we are the first to study the usefulness of dependency forests for relation extraction under a strong neural framework.

Graph neural network Graph neural networks (GNNs) have been successful in encoding dependency trees for downstream tasks, such as semantic role labeling (Marcheggiani and Titov, 2017), semantic parsing (Xu et al., 2018), machine translation (Song et al., 2019; Bastings et al., 2017), relation extraction (Song et al., 2018b) and sentence ordering (Yin et al., 2019). In particular, Song et al. (2018b) showed that GNNs are more effective than DAG networks (Peng et al., 2017) for modeling syntactic trees in relation extraction, which cause loss of important structural information. We are the first to exploit GNNs for encoding search spaces in the form of dependency forests.

3 Task

Formally, the input to our task is a sentence $\boldsymbol{s} = w_1, w_2, \ldots, w_N$, where N is the number of words in the sentence and w_i represents the i-th input word. $\boldsymbol{s}$ is annotated with boundary information ($\xi_1 : \xi_2$ and $\zeta_1 : \zeta_2$) of target entity mentions (ξ and ζ). We focus on the classic binary relation extraction setting (Quirk and Poon, 2017), where the number of associated mentions is two. The output is a relation from a predefined relation set $\boldsymbol{R} = (r_1, \ldots, r_M, \text{None})$, where "None" means that no relation holds for the entities.

Two steps are taken for predicting the correct relation given an input sentence. First, a dependency parser is used to label the syntactic structure of the input. Here our baseline system takes the standard approach, using the 1-best parser output tree $\boldsymbol{D}_T$ as features. In contrast, our proposed model uses the most confident parser forest $\boldsymbol{D}_F$ as features. Given $\boldsymbol{D}_T$ or $\boldsymbol{D}_F$, the second step is to encode both s and $\boldsymbol{D}_T/\boldsymbol{D}_F$ using a neural network, before making a prediction.

We make use of the same graph neural network encoder structure to represent dependency syntax information for both the baseline and our model. In particular, a graph recurrent neural network architecture (Beck et al., 2018; Song et al., 2018a; Zhang et al., 2018a) is used, which has been shown effective in encoding graph structures (Song et al., 2019), giving competitive results with alternative graph networks such as graph convolutional neural networks (Marcheggiani and Titov, 2017; Bastings et al., 2017).

4 Baseline: DepTree

As shown in Figure 2, our baseline model stacks a bidirectional LSTM layer to encode an input sentence $w_1, \ldots, w_N$ with a graph recurrent network

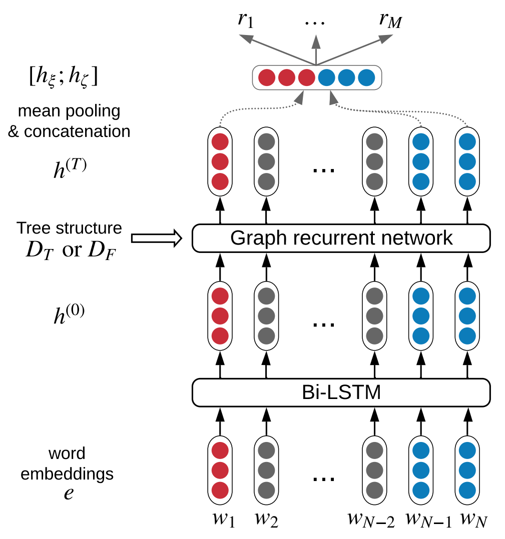


Figure 2: Framework of our baseline and model.

(GRN) to encode a 1-best dependency tree, which extracts features from the sentence and the dependency tree $\boldsymbol{D}_T$, respectively. Similar model frameworks have shown highly competitive performances in previous relation extraction studies (Peng et al., 2017; Song et al., 2018b).

4.1 Bi-LSTM layer

Given the input sentence $w_1, w_2, \ldots, w_N$, we represent each word with its embedding to generate a sequence of embeddings $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_N$. A Bi-LSTM layer is used to encode the sentences:

$$\overleftarrow{\boldsymbol{h}}_i^{(0)} = \text{LSTM}_l(\overleftarrow{\boldsymbol{h}}_{i+1}^{(0)}, \boldsymbol{e}_i) \quad (1)$$

$$\overrightarrow{\boldsymbol{h}}_i^{(0)} = \text{LSTM}_r(\overrightarrow{\boldsymbol{h}}_{i-1}^{(0)}, \boldsymbol{e}_i), \quad (2)$$

where the state of each word w_i is generated by concatenating the states of both directions:

$$\boldsymbol{h}_i^{(0)} = [\overleftarrow{\boldsymbol{h}}_i^{(0)}; \overrightarrow{\boldsymbol{h}}_i^{(0)}] \quad (3)$$

4.2 GRN layer

A 1-best dependency tree can be represented as a directed graph $\boldsymbol{D}_T = \langle \boldsymbol{V}, \boldsymbol{E} \rangle$, where $\boldsymbol{V}$ includes all words $w_1, w_2, \ldots, w_N$ and $\boldsymbol{E} = \{(w_j, l, w_i)\}|_{w_j \in V, w_i \in V}$ represents all dependency edges (Marcheggiani and Titov, 2017). Each triple (w_j, l, w_i) corresponds to a dependency edge, where w_j modifies w_i with an arc label l. Each word w_i is associated with a hidden state that is initialized with the Bi-LSTM output $\boldsymbol{h}_i^{(0)}$. The state representation of the entire tree consists of all word states:

$$\boldsymbol{h}^{(0)} = \{\boldsymbol{h}_i^{(0)}\}_{w_i \in V} \quad (4)$$

In order to capture non-local interactions between words, the GRN layer adopts a message passing framework that performs iterative information exchange between directly connected words. As a result, each word state is updated by absorbing larger contextual information through the message passing process, and a sequence of state transitions $\boldsymbol{h}^{(0)}, \boldsymbol{h}^{(1)}, \ldots$ is generated for the entire tree. The final state $\boldsymbol{h}^{(T)} = \text{GRN}(\boldsymbol{h}^{(0)}, T)$, where T is a hyperparameter representing the number of state transitions.

Message passing The message passing framework takes two main steps within each iteration: message calculation and state update. Take w_i and iteration t as the example. In the first step, separate messages $\boldsymbol{m}_i^{\uparrow}$ and $\boldsymbol{m}_i^{\downarrow}$ are calculated by summing up the messages of its children and parent in the dependency tree, respectively:

$$\boldsymbol{m}_i^{\uparrow} = \sum_{(w_j, l, w_i) \in \boldsymbol{E}_{(\cdot,\cdot,i)}} [\boldsymbol{h}_j^{(t-1)}; \boldsymbol{e}_l] \quad (5)$$

$$\boldsymbol{m}_i^{\downarrow} = \sum_{(w_i, l, w_k) \in \boldsymbol{E}_{(i,\cdot,\cdot)}} [\boldsymbol{h}_k^{(t-1)}; \boldsymbol{e}_{l_{rev}}], \quad (6)$$

where $\boldsymbol{E}_{(\cdot,\cdot,i)}$ and $\boldsymbol{E}_{(i,\cdot,\cdot)}$ represent all edges with a head word w_i and a modifier word w_i, respectively, and $\boldsymbol{e}_{l_{rev}}$ represents the embedding of label l_{rev}, the reverse version of original label l (such as "amod-rev" is the reverse version of "amod"). The message from a child or a parent is obtained by simply concatenating its hidden state with the corresponding edge label embedding.

In the second step, GRN uses standard gated operations of LSTM (Hochreiter and Schmidhuber, 1997) to update hidden state $\boldsymbol{h}_i^{(t-1)}$ with the previously integrated message. In particular, a cell $\boldsymbol{c}_i^{(t)}$ is taken to record memory for $\boldsymbol{h}_i^{(t)}$; an input gate $\boldsymbol{i}_i^{(t)}$, an output gate $\boldsymbol{o}_i^{(t)}$ and a forget gate $\boldsymbol{f}_i^{(t)}$ are used to control information flow from the inputs and to the output $\boldsymbol{h}_i^{(t)}$:

$$\begin{aligned}
\boldsymbol{i}_i^{(t)} &= \sigma(\boldsymbol{W}_1^{\uparrow}\boldsymbol{m}_i^{\uparrow} + \boldsymbol{W}_1^{\downarrow}\boldsymbol{m}_i^{\downarrow} + \boldsymbol{b}_1) \\
\boldsymbol{o}_i^{(t)} &= \sigma(\boldsymbol{W}_2^{\uparrow}\boldsymbol{m}_i^{\uparrow} + \boldsymbol{W}_2^{\downarrow}\boldsymbol{m}_i^{\downarrow} + \boldsymbol{b}_2) \\
\boldsymbol{f}_i^{(t)} &= \sigma(\boldsymbol{W}_3^{\uparrow}\boldsymbol{m}_i^{\uparrow} + \boldsymbol{W}_3^{\downarrow}\boldsymbol{m}_i^{\downarrow} + \boldsymbol{b}_3) \\
\boldsymbol{u}_i^{(t)} &= \tanh(\boldsymbol{W}_4^{\uparrow}\boldsymbol{m}_i^{\uparrow} + \boldsymbol{W}_4^{\downarrow}\boldsymbol{m}_i^{\downarrow} + \boldsymbol{b}_4) \\
\boldsymbol{c}_i^{(t)} &= \boldsymbol{f}_i^{(t)} \odot \boldsymbol{c}_i^{(t-1)} + \boldsymbol{i}_i^{(t)} \odot \boldsymbol{u}_i^{(t)} \\
\boldsymbol{h}_i^{(t)} &= \boldsymbol{o}_i^{(t)} \odot \tanh(\boldsymbol{c}_i^{(t)}),
\end{aligned} \quad (7)$$

where $\boldsymbol{W}_x^{\uparrow}$, $\boldsymbol{W}_x^{\downarrow}$, and $\boldsymbol{b}_x$ $(x \in \{1,2,3,4\})$ are model parameters, and $\boldsymbol{c}_i^{(0)}$ is initialized as a vector of zeros.

The same process repeats for T iterations. Starting from $\boldsymbol{h}^{(0)}$ of the Bi-LSTM layer, increasingly more informed hidden states $\boldsymbol{h}^{(t)}$ are obtained as the iteration increases, and $\boldsymbol{h}^{(T)}$ is used as the final representation of each word.

4.3 Relation prediction

Given $\boldsymbol{h}^{(T)}$ of the GRN encoding, we calculate the representation vector of the two related entity mentions ξ and ζ (such as "almorexant" and "orexin receptor" in Figure 1) with mean pooling:

$$\boldsymbol{h}_\xi = f_{\text{mean}}(\boldsymbol{h}^{(T)}_{\xi_1:\xi_2}) \tag{8}$$

$$\boldsymbol{h}_\zeta = f_{\text{mean}}(\boldsymbol{h}^{(T)}_{\zeta_1:\zeta_2}) \tag{9}$$

where $\xi_1 : \xi_2$ and $\zeta_1 : \zeta_2$ represent the span of ξ and ζ, respectively, and f_{mean} is the mean-pooling function.

Finally, the representations of both mentions are concatenated to be the input of a logistic regression classifier:

$$\boldsymbol{y} = \text{softmax}(\boldsymbol{W}_5[\boldsymbol{h}_\xi; \boldsymbol{h}_\zeta] + \boldsymbol{b}_5), \tag{10}$$

where $\boldsymbol{W}_5$ and $\boldsymbol{b}_5$ are model parameters.

5 Model

In this section, we first discuss how to generate high-quality dependency forests, before showing how to adapt GRN to consider the parser probability of each dependency edge.

5.1 Forest generation

Given a dependency parser, generating dependency forests with high recall and low noise is a non-trivial problem. On the one hand, keeping the whole search space gives 100% recall, but introduces maximum noise. On the other hand, using the 1-best dependency tree can result in low recall given an imperfect parser. We investigate two algorithms to generate high-quality forests by judging "quality" from different perspectives: one focusing on arcs, and the other focusing on trees.

EDGEWISE This algorithm focuses on the local relation of each individual edge and uses parser probabilities as confidence scores to assess edge qualities. Starting from the whole parser search space, it keeps all the edges with scores greater than a threshold γ. The time complexity is $O(N^2)$, where N represents the sentence length.[1]

KBESTEISNER This algorithm extends the Eisner algorithm (Eisner, 1996) with cube pruning (Huang and Chiang, 2005) for finding K highest-scored tree structures. The Eisner algorithm is a standard method for decoding 1-best trees for graph-based dependency parsing. Based on bottom-up dynamic programming, it stores the 1-best subtree for each span and takes $O(N^3)$ time complexity for decoding a sentence of N words.

KBESTEISNER keeps a sorted list of K-best hypotheses for each span. Cube pruning (Huang and Chiang, 2005) is adopted to generate the K-best list for each larger span from the K-best lists of its sub-spans. After the bottom-up decoding, we merge the final K-bests by combining identical dependency edges to make the forest. As a result, KBESTEISNER takes $O(N^3 K \log K)$ time.

Discussions EDGEWISE is much simpler and faster than KBESTEISNER. Compared with the $O(N^3 K \log K)$ time complexity of KBESTEISNER, EDGEWISE only takes $O(N^2)$ running time, and each step (storing an edge) runs faster than KBESTEISNER (making a new hypothesis by combining two from sub-spans). Besides, the forests of EDGEWISE can be denser and provide richer information than those from KBESTEISNER. This is because KBESTEISNER only merges K trees, where many edges are shared among them. Also, K cannot be set to a large number (such as 100), because that will cause a dramatic increase of running time.

Compared with KBESTEISNER, EDGEWISE suffers from two potential problems. First, EDGEWISE does not guarantee to produce a 1-best tree in a generated forest, as it makes decisions by considering the individual edges. Second, it does not guarantee to generate spanning forests, which can happen when the threshold γ is high. On the other hand, no previous work has shown that the information from the whole tree is crucial for relation extraction. In fact, many previous studies use only the dependency path between the target entity mentions (Bunescu and Mooney, 2005; Airola et al., 2008; Chowdhury et al., 2011; Gormley et al., 2015; Mehryary et al., 2016). We study

[1]More accurately, it is $O(N^2 L)$ and L s a *constant* factor, denoting the number of distinct dependency labels. We omit it for simplicity.

the effectiveness of both algorithms in our experiments.

5.2 GRN encoding with parser confidence

As illustrated by Figure 1(b), our dependency forests are directed graphs that can be consumed by GRN without any structural changes. For fair comparison, we use the same model as the baseline to encode sentences and forests. Thus our model uses **the same number of parameters** as our baseline taking 1-best trees.

Since forests contain more than one tree, it is intuitive to consider parser confidence scores for potentially better feature extraction. To this end, we slightly adjust the GRN encoding process without introducing additional parameters. In particular, we enhance the original message sum function (Equations 5 and 6) by applying the edge probabilities in calculating weighted message sums:

$$\boldsymbol{m}_i^{\uparrow} = \sum_{\epsilon \in \boldsymbol{E}_{(\cdot,\cdot,i)}} p_\epsilon [\boldsymbol{h}_j^{(t-1)}; \boldsymbol{e}_l] \tag{11}$$

$$\boldsymbol{m}_i^{\downarrow} = \sum_{\epsilon \in \boldsymbol{E}_{(i,\cdot,\cdot)}} p_\epsilon [\boldsymbol{h}_k^{(t-1)}; \boldsymbol{e}_{l_{rev}}], \tag{12}$$

where ϵ (instead of a triple) is used to represent an edge for simplicity, and p_ϵ is the parser probability for edge ϵ. The edge probabilities are *not* adjusted during end-task training.

6 Training

Relation loss Given a set of training instances, each containing a sentence $\boldsymbol{s}$ with two target mentions ξ and ζ, and a dependency structure $\boldsymbol{D}$ (tree or forest), we train our models with a cross-entropy loss between the gold-standard relations r and model distribution:

$$l_R = -\log p(r|\boldsymbol{s}, \xi, \zeta, \boldsymbol{D}; \boldsymbol{\theta}), \tag{13}$$

where $\boldsymbol{\theta}$ represents the model parameters.

Using additional NER loss For training on BioCreative VI CPR, we follow previous work (Liu et al., 2017; Verga et al., 2018) to take NER loss as additional supervision, though the mention boundaries are known during testing.

$$l_{NER} = -\frac{1}{N} \sum_{n=1}^{N} \log p(t_n|\boldsymbol{s}, \boldsymbol{D}; \boldsymbol{\theta}), \tag{14}$$

where t_n is the gold NE tag of w_n with the "BIO" scheme. Both losses are conditionally independent given the deep features produced by our model, and the final loss for BioCreative VI CPR training is $l = l_R + l_{NER}$.

7 Experiments

We conduct experiments on two medical benchmarks to test the usefulness of dependency forest.

7.1 Data

BioCreative VI CPR (Krallinger et al., 2017) This task[2] focuses on the relations between chemical compounds (such as drugs) and proteins (such as genes). The full corpus contains 1020, 612 and 800 extracted PubMed[3] abstracts for training, development and testing, respectively. All abstracts are manually annotated with the boundaries of entity mentions and the relations. The data provides three types of NEs: "CHEMICAL", "GENE-Y" and "GENE-N", and the relation set $\boldsymbol{R}$ contains 5 regular relations ("CPR:3", "CPR:4", "CPR:5", "CPR:6" and "CPR:9") and the "None" relation.

For efficient generation of dependency structures, we segment each abstract into sentences, keeping only the sentences that contain at least a chemical mention and a protein mention. For any sentence containing several chemical mentions or protein mentions, we keep multiple copies of it with each copy having different target mention pairs. As a result, we only consider the relations of mentions in the same sentence, assigning all cross-sentence chemical-protein pairs as "None" relation. By doing this, we effectively sacrifice cross-sentence relations, which has a negative effect on our systems; but this is necessary for efficient generation of dependency structures since directly parsing a short paragraph is slow and erroneous.[4] In general, we obtain 16,107 training, 10,030 development and 14,269 testing instances, in which around 23% have regular relations. The highest recalls for relations on our development and test sets are 92.25 and 92.54, respectively, because of the exclusion of cross-sentence relations in preprocessing. We report F1 scores of the full test set for a fair comparison, using all gold regular relations to calculate recalls.

Phenotype-Gene relation (PGR) (Sousa et al., 2019) This dataset concerns the relations between

[2]https://biocreative.bioinformatics.udel.edu/tasks/biocreative-vi/track-5/

[3]https://www.ncbi.nlm.nih.gov/pubmed/

[4]Peng et al. (2017) describe a solution for cross-sentence cases, which joins different dependency structures by connecting their roots. We leave it for future work.

human phenotypes (such as diseases) with human genes, where the relation set is a *binary* class on whether a phenotype is related to a gene. It has 18,451 silver training instances and 220 high-quality test instances, with each containing mention boundary annotations. We separate the first 15% training instances as our development set. Unlike *BioCreative VI CPR*, almost every relation of PGR is within a single sentence.

7.2 Models

We compare the following models:

- TEXTONLY: It does not take dependency structures and directly uses the Bi-LSTM outputs ($\boldsymbol{h}^{(0)}$ in Eq. 3) to make predictions.
- DEPTREE: Our baseline using 1-best dependency trees, as shown in Section 4.
- EDGEWISEPS and EDGEWISE: Our models using the forests generated by our EDGEWISE algorithm with or without parser scores.
- KBESTEISNERPS and KBESTEISNER: Our model using the forests generated by our KBESTEISNER algorithm with or without parser scores, respectively.

7.3 Settings

We take a state-of-the-art deep biaffine parser (Dozat and Manning, 2017), trained on the Penn Treebank (PTB) (Marcus and Marcinkiewicz, 1993) converted to Universal Dependency, to obtain 1-best trees and full search spaces for generating forests. Using standard PTB data split (02–21 for training, 22 for development and 23 for testing), it gives UAS and LAS scores of 95.7 and 94.6, respectively.

For the other hyper-parameters, word embeddings are initialized with the 200-dimensional BioASQ vectors[5], pretrained on 10M abstracts of biomedical articles, and are fixed during training. The dimension of hidden vectors in Bi-LSTM is set to 200, and the number of message passing steps T is set to 2 based on Zhang et al. (2018b). We use Adam (Kingma and Ba, 2014), with a learning rate of 0.001, as the optimizer. The batch size, coefficient for $l2$ normalization loss and dropout rate are 20, 10^{-8} and 0.1, respectively.

[5]http://bioasq.lip6.fr/tools/BioASQword2vec/

γ	#Edge/#Node	LAS	Conn. Ratio(%)
0.05	2.09	92.5	100.0
0.1	1.57	91.2	99.5
0.2	1.34	90.5	94.2
0.3	1.04	88.0	77.6
K	#Edge/#Node	LAS	Conn. Ratio(%)
1	1.00	86.4	100.0
2	1.03	87.3	100.0
5	1.09	89.1	100.0
10	1.14	89.8	100.0

Table 1: Statistics on forests generated with various γ (upper half) and K (lower half) on the development set.

7.4 Analyses of generated forests

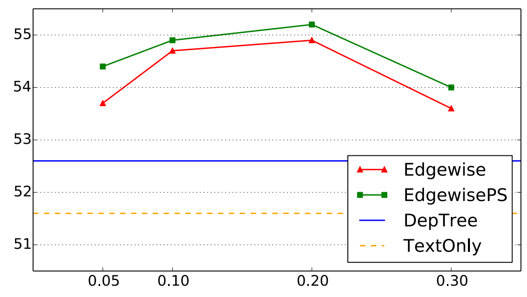
Table 1 demonstrates several characteristics of the generated forests of both the EDGEWISE and KBESTEISNER algorithms in Section 5.1, where "#Edge/#Sent" measures the forest density with the number of edges divided by the sentence length, "LAS" represents the oracle LAS score on 100 biomedical sentences with manually annotated dependency trees, and "Conn. Ratio (%)" shows the percentage of forests where both related entity mentions are connected.

Regarding the forest density, forests produced by EDGEWISE generally contain more edges than those from KBESTEISNER. Due to the combinatorial property of forests, EDGEWISE can give much more candidate trees (and sub-trees) for the whole sentence (and each sub-span). This coincides with the fact that the forests generated by EDGEWISE have higher oracle scores than these generated by KBESTEISNER.

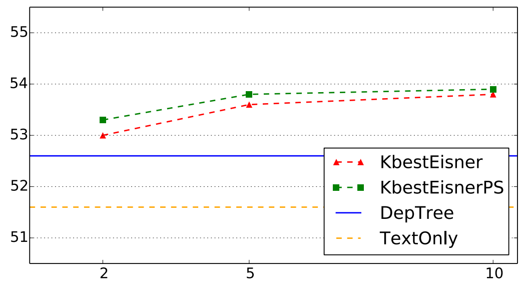
For connectivity, KBESTEISNER guarantees to generate spanning forests. On the other hand, the connectivity ratio for the forests produced by EDGEWISE drops when increasing the threshold γ. We can have more than 94% being connected with $\gamma \leq 0.2$. Later we will show that good end-task performance can still be achieved with the 94% connectivity ratio. This indicates that losing connectivity for a small potion of the data may not hurt the overall performance.

7.5 Development results

Figure 3 shows the development experiments for our forest generation algorithms, where both EDGEWISE and KBESTEISNER give consistent



(a) EDGEWISE



(b) KBESTEISNER

Figure 3: Development results (F1 score) for our forest generation methods.

improvements over DEPTREE and TEXTONLY. Generally, EDGEWISE gives more improvements than KBESTEISNER. The main reason may be that EDGEWISE generates denser forests, providing richer features. On the other hand, KBESTEISNER shows a marginal improvement by increasing K from 5 to 10. This indicates that only merging 10-best trees may be far from sufficient. However, using a much larger K (such as 100) is not practical due to dramatically increased computation time. In particular, the running time of KBESTEISNER with $K = 10$ is already much longer than that of EDGEWISE. As a result, EDGEWISE better serves our goal compared to KBESTEISNER. This may sound surprising, as EDGEWISE does not consider tree-level scores. It suggests that relation extraction may not require full dependency tree features. This coincides with previous relation extraction research (Bunescu and Mooney, 2005; Airola et al., 2008), which utilizes the shortest path connecting the two candidate entities in the dependency tree.

Leveraging parser confidence scores also consistently helps both methods. It is especially effective for EDGEWISE when $\gamma = 0.05$. This is likely because the parser confidence scores are useful for distinguishing some erroneous dependency arcs, when noise is large (e.g. when γ is too small). Following the development results, we directly report the performances of EDGEWISEPS and KBESTEISNERPS, setting γ and K to 0.2 and 10, respectively, in our remaining experiments.

Model	F1 score
GRU+Attn (Liu et al., 2017)†	49.5
Bran (Verga et al., 2018)†	50.8
TEXTONLY	50.6
DEPTREE	51.4
KBESTEISNERPS	52.4**
EDGEWISEPS	**53.4****

Table 2: Test results of Biocreative VI CPR. † indicates previously reported numbers. ** means significant over DEPTREE at $p < 0.01$ with 1000 bootstrap tests (Efron and Tibshirani, 1994).

7.6 Main results on BioCreative VI CPR

Table 2 shows the main comparison results on the BioCreative CPR testset, with comparisons to the previous state-of-the-art and our baselines. *GRU+Attn* (Liu et al., 2017) stacks a self-attention layer on top of GRU (Cho et al., 2014) and embedding layers; *Bran* (Verga et al., 2018) adopts a biaffine self-attention model to simultaneously extract the relations of all mention pairs. Both methods use only textual knowledge.

TEXTONLY gives a performance comparable with *Bran*. With 1-best dependency trees, our DEPTREE baseline gives better performances than the previous state of the art. This confirms the usefulness of dependency structures and the effectiveness of GRN on encoding these structures. Using dependency forests and parser confidence scores, both KBESTEISNERPS and EDGEWISEPS obtain significantly higher numbers than DEPTREE. Consistent with the development experiments, EDGEWISEPS has a higher testset performance than KBESTEISNERPS.

7.7 Analysis

Effectiveness on parsing accuracy We have shown in Sections 7.5 and 7.6 that a dependency parser trained using a domain-general treebank can produce high-quality dependency forests in a target domain (biomedical) for helping relation extraction. This is based on the assumption of there being a high-quality treebank in a descent scale, which may not be true for low-resource languages. We simulate this low-resource effect by training our parser in much smaller treebanks of 1K or 5K

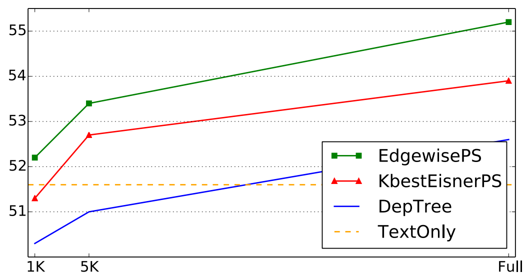


Figure 4: DEV results of BioCreative CPR regarding the dependency parsers trained on different number (1K, 5K or Full) of dependency trees.

dependency trees, respectively. The LAS scores for the resulting parsers on our 100 manually annotated biomedical dependency trees are 79.3 and 84.2, respectively, while the LAS score for the parser trained with the full treebank is 86.4, as shown in Table 1.

Figure 4 shows the results on the Biocreative CPR development set, where the performance of TEXTONLY is 51.6. DEPTREE fails to outperform TEXTONLY when only 1K or 5K dependency trees are available for training our parser. This is due to the low parsing recall and subsequent noise caused by the weak parsers. It confirms the previous conclusion that dependency structures are highly influential to the performance of relation extraction. Both EDGEWISEPS and KBESTEISNERPS are still more effective than DEPTREE. In particular, KBESTEISNERPS significantly improves TEXTONLY with 5K dependency trees, and EDGEWISEPS is helpful even with 1K dependency trees.

KBESTEISNER shows relatively smaller gaps than EDGEWISE when only a limited number of dependency trees are available. This is probably because considering whole-tree quality helps to better eliminate noise.

Case study Figure 5 illustrates two major types of errors in BioCreative CPR, which are caused by inaccurate 1-best dependency trees. As shown in Figure 5(a), the baseline system mistakenly predicts a "None" relation for that instance. This is mainly because "STAT3" is incorrectly linked to the main verb "inhibited" with a "punct" relation, but it should be linked to "AKT". In contrast, our forest contains the correct relation and with a probability of 0.18. This is possibly because "AKT and STAT3" fits the common pattern of "A and B" that conjunct two nouns.

Figure 5(b) shows another type of parsing er-

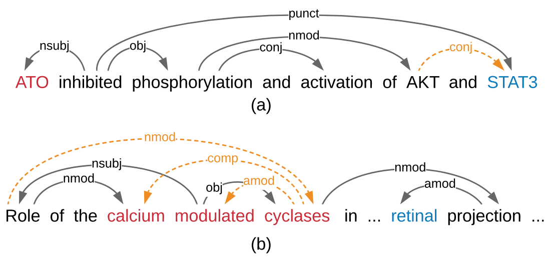

punct
nsubj
obj
nmod
conj
conj
ATO inhibited phosphorylation and activation of AKT and STAT3
(a)
nmod
nsubj
comp
nmod
obj
amod
nmod
amod
Role of the calcium modulated cyclases in ... retinal projection ...
(b)

Figure 5: Two representative cases in BioCreative CPR, contrasting 1-best trees and forests, where irrelevant content and arcs are omitted for simplicity.

Model	F1 score
BO-LSTM (Lamurias et al., 2019)†	52.3
BioBERT (Lee et al., 2019)†	67.2
TEXTONLY	76.0
DEPTREE	78.9
KBESTEISNERPS	83.6*
EDGEWISEPS	**85.7****

Table 3: Main results on PGR testest. † denotes previous numbers rounded into 3 significant digits. * and ** indicate significance over DEPTREE at $p < 0.05$ and $p < 0.01$ with 1000 bootstrap tests.

rors that cause end-task mistakes. In this example, the multi-token mention "calcium modulated cyclases" is incorrectly segmented in the 1-best dependency tree, where "modulated" is used as the main verb of the whole sentence, leaving "cyclases" and "calcium" as the object and the modifier of the subject, respectively. However, this mention ought to be a noun phrase with "cyclases" being the head. Our forest helps in this case by providing a more reasonable structure (shown as the yellow dashed arcs), where both "calcium" and "modulated" modify "cyclases". This is likely because "modulated" can be interpreted as an adjective in addition to being a verb. It shows the advantage of keeping multiple candidate syntactic arcs.

7.8 Main results on PGR

Table 3 shows the comparison with previous work on the PGR testset, where our models are significantly better than the existing models. This is likely because the previous models do not utilize all the information from inputs: *BO-LSTM* only takes the words (without arc labels) along the shortest dependency path between the target mentions; the pretrained weights of *BioBERT* are kept constant during training for relation extraction.

With 1-best trees, DEPTREE is 2.9 points bet-

Model	F1 score
C-GCN (Zhang et al., 2018b)†	84.8
C-AGGCN (Guo et al., 2019)†	85.7
DEPTREE	84.6
KBESTEISNERPS	85.8
EDGEWISEPS	86.3

Table 4: Main results on SemEval-2010 task 8 testest. † denotes previous numbers.

ter than TEXTONLY, confirming the usefulness of dependency structures. Leveraging dependency forests, both KBESTEISNERPS and EDGEWISEPS significantly outperform DEPTREE with p-values of 0.003 and 0.024, respectively. This further confirms the usefulness of dependency forests for medical relation extraction.

7.9 Main results on SemEval-2010 task 8

In addition to the biomedical domain, leveraging dependency forests applies to other domains as well. As shown in Table 4, we conduct a preliminary study on SemEval-2010 task 8 (Hendrickx et al., 2009), a widely used benchmark for news-domain relation extraction. It is a public dataset, containing 10,717 instances (8000 for training and development, 2717 for testing) with 19 relations: 9 directed relations and a special "Other" class. Both *C-GCN* and *C-AGGCN* take a similar network as ours by stacking a graph neural network for encoding trees on top of a Bi-LSTM layer for encoding sentences.

DEPTREE achieves similar performance as *C-GCN* and is slightly worse than *C-AGGCN*, with one potential reason being that *C-AGGCN* takes more parameters. Using forests, both KBESTEISNERPS and EDGEWISEPS outperform DEPTREE with the same number of parameters, and they show comparable and slightly better performances than *C-AGGCN*. Again, EDGEWISEPS is better than KBESTEISNERPS, showing that the former is a better way for generating forests.

8 Conclusion

We have proposed two algorithms for generating high-quality dependency forests for relation extraction, and studied a graph recurrent network for effectively distinguishing useful features from noise in parsing forests. Experiments on two biomedical relation extraction benchmarks show the superiority of forests versus tree structures, without introducing any additional model parameters. Our deep analyses indicate that the main advantage comes from alleviating out-of-domain parsing errors.

Acknowledgments Research supported by NSF award IIS-1813823.

References

Asma Ben Abacha and Pierre Zweigenbaum. 2011. Automatic extraction of semantic relations between medical entities: a rule based approach. *Journal of biomedical semantics*, 2(5).

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. 2008. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics*, 9(11):S2.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*.

Marie Candito, Enrique Henestroza Anguiano, and Djamé Seddah. 2011. A word clustering approach to domain adaptation: Effective parsing of biomedical texts. In *Proceedings of the 12th International Conference on Parsing Technologies*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Faisal Md. Chowdhury, Alberto Lavelli, and Alessandro Moschitti. 2011. A study on dependency tree kernels for automatic extraction of protein-protein interaction. In *Proceedings of BioNLP 2011 Workshop*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of International Conference on Learning Representations*.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*.

Carol Friedman, Pauline Kra, Hong Yu, Michael Krauthammer, and Andrey Rzhetsky. 2001. Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17(1).

Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1774–1784.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of SemEval*, pages 94–99.

Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. 2005. Overview of biocreative: critical assessment of information extraction for biology.

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

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Martin Krallinger, Obdulia Rabal, Saber A Akhondi, et al. 2017. Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the VI BioCreative challenge evaluation workshop*.

Andre Lamurias, Diana Sousa, Luka A Clarke, and Francisco M Couto. 2019. BO-LSTM: classifying relations via long short-term memory networks along biomedical ontologies. *BMC bioinformatics*, 20(1):10.

Phong Le and Willem Zuidema. 2015. The forest convolutional network: Compositional distributional semantics with a neural chart and without binarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Matthew Lease and Eugene Charniak. 2005. Parsing biomedical literature. In *International Conference on Natural Language Processing*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: pre-trained biomedical language representation model for biomedical text mining. *arXiv preprint arXiv:1901.08746*.

Jiangming Liu and Yue Zhang. 2017. In-order transition-based constituent parsing. *Transactions of the Association for Computational Linguistics*, 5.

Sijia Liu, Feichen Shen, Yanshan Wang, Majid Rastegar-Mojarad, Ravikumar Komandur Elayavilli, Vipin Chaudhary, and Hongfang Liu. 2017. Attention-based neural networks for chemical protein relation extraction. In *Proceedings of the BioCreative VI Workshop*.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng WANG. 2015. A dependency-based neural network for relation classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*.

Wei Lu and Hwee Tou Ng. 2011. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Chunpeng Ma, Akihiro Tamura, Masao Utiyama, Tiejun Zhao, and Eiichiro Sumita. 2018. Forest-based neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Mitchell P Marcus and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2).

David McClosky and Eugene Charniak. 2008. Self-training for biomedical parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.

Farrokh Mehryary, Jari Björne, Sampo Pyysalo, Tapio Salakoski, and Filip Ginter. 2016. Deep learning with minimal training data: TurkuNLP entry in the BioNLP shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proceedings of ACL-08: HLT*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

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.

Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL-17)*.

Kenji Sagae, Yusuke Miyao, Rune Sætre, and Jun'ichi Tsujii. 2008. Evaluating the effects of treebank size in a practical application for parsing. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*.

Parikshit Sondhi, Manish Gupta, ChengXiang Zhai, and Julia Hockenmaier. 2010. Shallow information extraction from medical forum data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using amr. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018a. A graph-to-sequence model for amr-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018b. N-ary relation extraction using graph-state lstm. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2226–2235.

Diana Sousa, André Lamúrias, and Francisco M Couto. 2019. A silver standard corpus of human phenotype-gene relations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Zhaopeng Tu, Yang Liu, Young-Sook Hwang, Qun Liu, and Shouxun Lin. 2010. Dependency forest for statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Hua Xu, Shane P Stenner, Son Doan, Kevin B Johnson, Lemuel R Waitman, and Joshua C Denny. 2010. Medex: a medication information extraction system for clinical narratives. *Journal of the American Medical Informatics Association*, 17(1).

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015a. Semantic relation classification via convolutional neural networks with simple negative sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Kun Xu, Lingfei Wu, Zhiguo Wang, Mo Yu, Liwei Chen, and Vadim Sheinin. 2018. Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015b. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Yongjing Yin, Linfeng Song, Jinsong Su, Jiali Zeng, Chulun Zhou, and Jiebo Luo. 2019. Graph-based neural sentence ordering. In *Proceedings of IJCAI*.

Hong Yu and Eugene Agichtein. 2003. Extracting synonymous gene and protein terms from biological literature. *Bioinformatics*, 19.

Poorya Zaremoodi and Gholamreza Haffari. 2018. Incorporating syntactic uncertainty in neural machine translation with a forest-to-sequence model. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1421–1429.

Yue Zhang, Qi Liu, and Linfeng Song. 2018a. Sentence-state lstm for text representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 317–327.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018b. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.