Improving Relation Extraction through Syntax-induced Pre-training with Dependency Masking

Yuanhe Tian[♥], **Yan Song**[♠†], **Fei Xia**[♥]
[♥]University of Washington [♠]The Chinese University of Hong Kong (Shenzhen)
[♥]{yhtian, fxia}@uw.edu [♠]songyan@cuhk.edu.cn

Abstract

Relation extraction (RE) is an important natural language processing task that predicts the relation between two given entities, where a good understanding of the contextual information is essential to achieve an outstanding model performance. Among different types of contextual information, the auto-generated syntactic information (namely, word dependencies) has shown its effectiveness for the task. However, most existing studies require modifications to the existing baseline architectures (e.g., adding new components, such as GCN, on the top of an encoder) to leverage the syntactic information. To offer an alternative solution, we propose to leverage syntactic information to improve RE by training a syntax-induced encoder on auto-parsed data through dependency masking. Specifically, the syntax-induced encoder is trained by recovering the masked dependency connections and types in first, second, and third orders, which significantly differs from existing studies that train language models or word embeddings by predicting the context words along the dependency paths. Experimental results on two English benchmark datasets, namely, ACE2005EN and SemEval 2010 Task 8 datasets, demonstrate the effectiveness of our approach for RE, where our approach outperforms strong baselines and achieve state-of-the-art results on both datasets.[1]

1 Introduction

Relation extraction (RE) provides deep analyses of the input text by extracting the relation between two given entities in the input. Therefore, it is an important task in natural language processing (NLP) and is widely used in many downstream NLP applications such as summarization (Wang and Cardie, 2012), question answering systems (Xu et al., 2016) and text mining (Distiawan et al., 2019). To correctly extract the relation between two entities, it normally requires a good modeling and analysis of the input text. Recent models such as LSTM, Transformers (Vaswani et al., 2017), and pre-trained language models (e.g., BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019)) have significantly improved the performance of RE models with an important reason of their encoding power on contextual information. However, such models still reach a bottleneck because it is hard for them to capture structural information of the running text (which is essential for RE) by modeling the text as a linear sequence of words. To deal with this situation, extra knowledge and features (e.g., syntactic knowledge) are used in many studies, while of all choices, the dependency parses have been widely used and demonstrated to be effective (Xu et al., 2015; Zhang et al., 2018; Guo et al., 2019; Mandya et al., 2020; Sun et al., 2020; Yu et al., 2020b; Tian et al., 2021), for the reason that the dependency trees are able to provide long-distance word-word relations which are important structural complement to existing models for RE.

To leverage dependency information, most existing approaches in NLP either treat it as extra input features (Prokopidis and Papageorgiou, 2014; Kiperwasser and Goldberg, 2015; Yu and Bohnet, 2017), which requires heavy feature engineering, or use complicated architectures (Xu et al., 2015; Roth and Lapata, 2016; Marcheggiani and Titov, 2017; Zhang et al., 2018; Li et al., 2018; Guo et al., 2019; Nie et al., 2020; Li et al., 2020a,b; Chen et al., 2020) to encode it, which suffers from the difficulty of designing an effective model. In addition, these approaches normally require dependency trees as extra input when processing sentences, and thus potentially suffer from noises from the dependency trees because of errors from automatic parsing. Therefore, an alternative is needed to leverage dependency information, especially auto-generated

[†]Corresponding author.

[1]The code involved in this paper are released at https://github.com/synlp/RE-DMP.

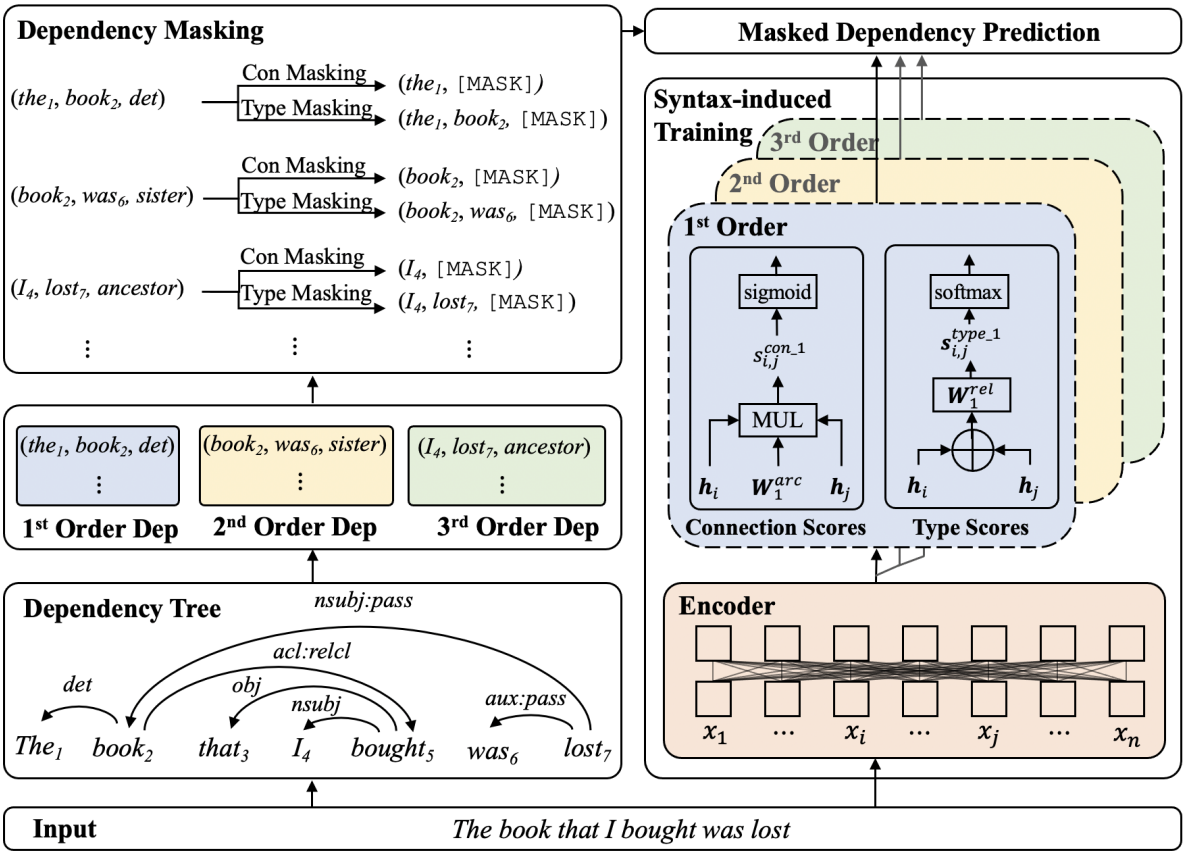

Figure 1: An overview of our approach to train a syntax-induced encoder (highlighted in the red box). The left part shows the process to extract and mask dependencies (connection and type masking, respectively) in first, second, and third orders, where the word subscript denotes its sentential index. The right part illustrates the process to compute the scores of dependency connections and types in different orders to recover the ones that being masked.

ones, for NLU tasks, so as to overcome the aforementioned issues.

In this paper, we propose to enhance RE through learning a good encoder equipped with dependency information, where the learning is carried out by a dependency-guided process. In detail, a dependency masking approach is designed to introduce such information, where we firstly apply an off-the-shelf dependency parser to large raw data and extract the dependency connections and types from the auto-parsed dependency trees, and then mask these connections and types so as to pre-train a syntax-induced encoder by recovering (predicting) them, which significantly differs from that of training word embeddings (Levy and Goldberg, 2014; Komninos and Manandhar, 2016) by predicting the context words along the dependency relations. In doing so, the dependency information weakly supervises the encoder and the pre-training on dependency masking ensures a selective learning process on those frequent and important dependency relations, which is more flexible than taking dependency parses (with noises) as fixed knowledge. In addition, by noting that higher order dependency information is beneficial in many cases (Coppola and Steedman, 2013; Kamigaito et al., 2018; Li et al., 2020b), we further enhance our approach by pre-training with masking second and third order word dependencies rather than just doing it on the first order. Once pre-trained, the resulted encoder is applied with ordinary fine-tune procedure for RE. Experimental results on two English benchmark datasets, namely, English ACE2005EN[2] and SemEval 2010 Task 8 (Hendrickx et al., 2010), for RE demonstrate the effectiveness of our approach, which outperforms strong baselines and achieves state-of-the-art results on both of the datasets.

2 The Approach

To learn a text encoder with important structural information for RE, we propose to pre-train it with masking and recovering word-word dependency connections and types that are auto-analyzed from

[2] https://catalog.ldc.upenn.edu/LDC2006T06

existing parsers. The resulted syntax-induced encoder is thus weakly supervised by such information and provided with necessary syntax integration. In doing so, the dependency information is introduced during pre-training the encoder, thus no extra input is required in applying it to real applications, avoiding particular design of models to leverage such information during inference. Figure 1 illustrates the architecture of our approach to learn from an input sentence $\mathcal{X} = x_1 x_2 \cdots x_i \cdots x_j \cdots x_n$ with n words and its dependency tree $\mathcal{T}_X$, so that the masking and recovering can be formalized by

$$Y^*_{\mathrm{M}} = \mathcal{DM}(\mathcal{DE}(\mathcal{T}_X)) \quad (1)$$

and

$$\widehat{Y}_{\mathrm{M}} = f(\mathcal{EN}(\mathcal{X})) \quad (2)$$

respectively, where Y^*_{M} is the set of all masked dependency connections and types obtained by dependency extraction ($\mathcal{DE}$) and dependency masking ($\mathcal{DM}$), and f the process (with pre-training on it) to recover (predict) Y^*_{M} to $\widehat{Y}_{\mathrm{M}}$, with the base encoder $\mathcal{EN}$ trained accordingly during the process. In the following text, we firstly illustrate dependency extraction, then the process to integrate syntax information into the encoder with dependency masking, and finally the steps to apply the resulted syntax-induced encoder to RE.

2.1 Dependency Extraction

To extract dependency information form the input text, we firstly apply an off-the-shelf dependency parser to the input and obtain its dependency tree $\mathcal{T}_X$. Then, we extract first, second, and third order[3] dependency information from $\mathcal{T}_X$ and represent them in the form of tuple, i.e., $(x_i, x_j, type)$, where there is a connection between x_i and x_j and the dependency type (which is directional) of x_j towards x_j is $type$. Specifically, for the first order dependencies, we directly use the dependency connections and types in $\mathcal{T}_X$, where we construct a directed connection between x_i and x_j (denoted by (x_i, x_j)) if x_j is the head of x_i and the dependency type between them is the syntactic role (e.g., nominal subject) of x_i with respect to x_j. For the second order dependencies, we construct a second order dependency connection between x_i and x_j if there is a word x' that connects to both x_i and x_j by two connections (x_i, x') and (x', x_j) in $\mathcal{T}_X$. In the second order case, we define three types for their connections namely, *ancestor*, *sister*, and *descendant*, according to the position of x_i and x_j in the dependency tree $\mathcal{T}_X$, which are illustrated in (I), (II), and (III) in Figure 2 (a), respectively.[4] Similarly, for third order dependencies, we extend the types to four ones, namely, *ancestor*, *uncle*, *nephew*, and *descendant*, which are illustrated in (I)-(IV) in Figure 2 (b).

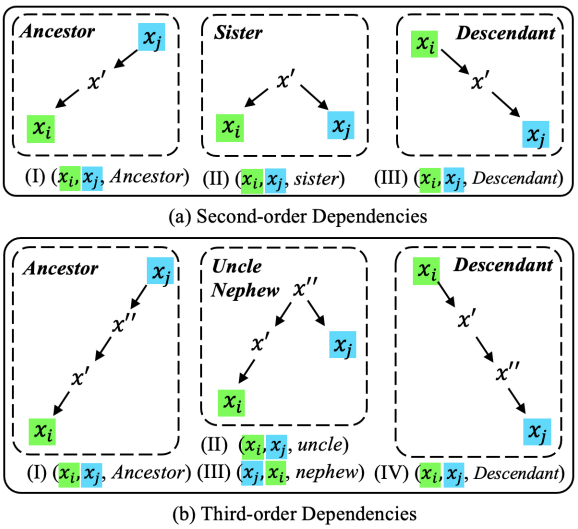

Figure 2: An illustration of the three second-order (a) and four third-order (b) dependency types between x_j and x_i, based on their positions in the parse tree.

2.2 Dependency Masking and Prediction

Previous studies leveraging dependency information by pre-training mainly focused on predicting the context words associated through dependency connections. Compared with these approaches, ours focuses on a different direction to leverage auto-parsed dependency information through learning word-word associations (i.e., dependency connections) and their dependency types. In doing so, we propose a weakly supervised learning task, namely, dependency masking (DM) with masked dependency prediction (MDP), to enhance text encoder pre-training, where they are paired processes that DM masks all connections and dependency types associated with each x_i (the masked connections and relations are denoted by $(x_i, \texttt{[MASK]})$ and $(x_i, x_j, \texttt{[MASK]})$ in Figure 1, respectively) and MDP aims to recover them during training.

[3] Most previous studies (Coppola and Steedman, 2013; Ji et al., 2019; Li et al., 2020b) use second or third order dependencies and some (Kamigaito et al., 2018) try higher orders yet show comparable performance.

[4] One can directly combine the dependency types of the connections (x_i, x') and (x', x_j) to represent such type for this scenario, but there will be huge numbers of combinations of syntactic roles, potentially leading to overfitting.

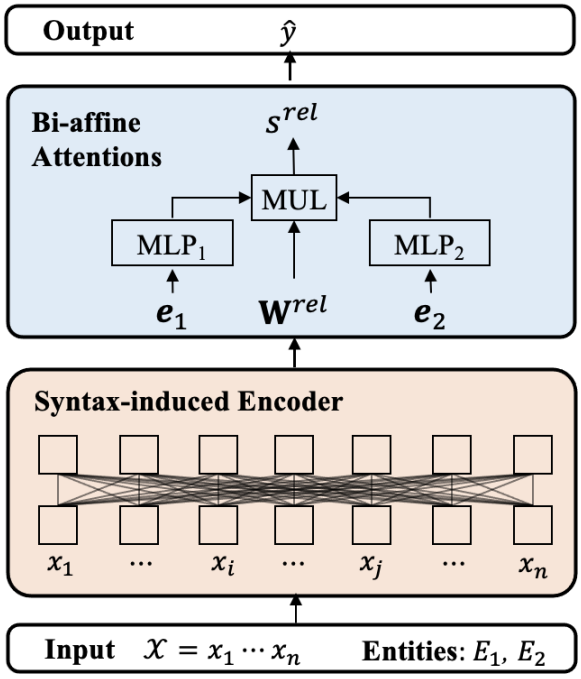

Figure 3: The architecture of our model for RE with the syntax-induced encoder (highlighted in red color) pre-trained on auto-parsed data through DMP.

Specifically, to recover the masked dependency connections and types, we firstly pass the input $\mathcal{X}$ into the base encoder (shown in the red box in Figure 1) that can be initialized in different ways (e.g., by a pre-trained language model) and obtain the hidden vector $\mathbf{h}_i$ of the word x_i. Then, we use three modules with the same architecture to recover masked dependency connections and types from first, second, and third order dependencies. Taking the first order dependencies as examples, we compute the connection score $s^{con\_1}_{i,j}$ and type scores $\mathbf{s}^{type\_1}_{i,j}$ for each pair of x_i and x_j by

$$s^{con\_1}_{i,j} = \mathbf{h}_j^\top \cdot \mathbf{W}_1^{con} \cdot \mathbf{h}_i \quad (3)$$

$$\mathbf{s}^{type\_1}_{i,j} = \mathbf{W}_1^{type} \cdot (\mathbf{h}_i \oplus \mathbf{h}_j) \quad (4)$$

where $\oplus$ denotes vector concatenation; $\mathbf{W}_1^{con}$ and $\mathbf{W}_1^{type}$ are trainable matrices. Herein, $s^{con\_1}_{i,j}$ is a scalar and $\mathbf{s}^{type\_1}_{i,j}$ is a vector with the values representing the scores for all possible types between x_i and x_j. Similarly, we use the same procedure to obtain the connection scores $s^{con\_2}_{i,j}$, $s^{con\_3}_{i,j}$ and the type score vectors $\mathbf{s}^{type\_2}_{i,j}$, $\mathbf{s}^{type\_3}_{i,j}$ for second and third order dependencies, respectively. Based on the connection and type scores of the first, second, and third order dependencies, our model recovers the masked connection by treating it as a binary classification using `sigmoid` function and predicts the masked type by applying `softmax` to the type score vectors. As a result, dependency information in different orders is implicitly introduced into the base encoder by the gradients back-propagated from the MDP process.

2.3 RE with Syntax-induced Encoder

Once the encoder is trained, we extract the obtianed syntax-induced encoder and fine-tune it on RE tasks, where the goal of our RE model is to predict the relation $\widehat{y} \in \mathcal{R}$ ($\mathcal{R}$ is the set for all relation types) between two given entities E_1 and E_2 in the input $\mathcal{X}$, which is formally expressed by

$$\widehat{y} = \arg\max_{rel \in \mathcal{R}} \; s\left(rel | \left(\mathcal{X}, E_1, E_2\right)\right) \quad (5)$$

where $s(\cdot)$ computes the score s^{rel} for a particular relation type $rel \in \mathcal{R}$ with the given input $\mathcal{X}$ and entities (i.e., E_1 and E_2). In doing so, we firstly fed $\mathcal{X}$ into the pre-trained syntax-induced encoder and obtain the hidden vectors $\mathbf{h}_i$ for each x_i. Next, we apply the max pooling operation to the hidden vectors of the words in each entity and obtain the vector representations, namely, $\mathbf{e}_1$ and $\mathbf{e}_1$, of E_1 and E_2. Then, we apply bi-affine attentions (Vaswani et al., 2017) to $\mathbf{e}_1$ and $\mathbf{e}_2$ to compute the score s^{rel} for the particular relationship rel. Specifically, bi-affine attentions pass $\mathbf{e}_1$ and $\mathbf{e}_2$ into two different multi-layer perceptrons (MLP), namely, MLP_1 and MLP_2, and use a trainable relationship matrix $\mathbf{W}^{rel}$ to compute s^{rel} via

$$\mathbf{e}'_1 = \text{MLP}_1(\mathbf{e}_1) \quad (6)$$

$$\mathbf{e}'_2 = \text{MLP}_2(\mathbf{e}_2) \quad (7)$$

$$s^{rel} = (\mathbf{e}'_1 \oplus [1])^\top \cdot \mathbf{W}^{rel} \cdot (\mathbf{e}'_2 \oplus [1]) \quad (8)$$

where $[1]$ is a one-dimensional unit vector which is the bias term for $\mathbf{e}'_1$ and $\mathbf{e}'_2$. Afterwards, we compute the scores s^{rel} for all types of relations and predict the one with the highest score.

3 Experimental Settings

3.1 Datasets

We use the newest English Wikipedia dump (Wiki) as the raw data to train the syntax-induced encoder through masked dependency prediction (MDP). We filter out sentences whose lengths are fewer than 10 words and obtain the resulting corpus with 92M sentences and 2,380M tokens. In obtaining dependency relations, we use Berkeley Neural Parser[5]

[5]We obtain their models from https://github.com/nikitakit/self-attentive-parser.

Datasets		Sent. #	Token #	Instance #
ACE05	Train	7K	145K	5K
	Dev	2K	36K	1K
	Test	2K	31K	1K
SemEval	Train	8K	141K	8K
	Test	3K	48K	3K

Table 1: The statistics of the two English benchmark datasets used in our experiments for relation extraction, where the number of sentence, tokens, and instances (i.e., entity pairs) are reported.

(Kitaev and Klein, 2018) trained on English Penn Treebank (PTB)[6] (Marcus et al., 1993) to automatically parse the Wiki data into constituency trees and then convert them into dependency trees by Stanford Dependency converter[7] (Manning et al., 2014). For relation extraction, we use English ACE2005EN (ACE05)[8] and SemEval 2010 Task 8 (SemEval)[9] (Hendrickx et al., 2010) with the standard train/dev/test splits[10] and follow previous studies (Christopoulou et al., 2018; Ye et al., 2019; Zhang et al., 2017; Soares et al., 2019) to process them. The statistics, namely, the number of sentences and tokens, as well as the number of instances (i.e., entity pairs), of both datasets are reported in Table 1.

3.2 Implementation Details

Since a good text representation plays an important role in achieving outstanding performance in many NLP tasks (Song and Shi, 2018; Devlin et al., 2019; Yang et al., 2019; Liu et al., 2019; Lewis et al., 2020; Song et al., 2021; Sun et al., 2021), in the experiments, we use pre-trained language models, i.e., BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019) that have demonstrate their effectiveness in many NLP tasks (Yan et al., 2020; Tian et al., 2020; Ke et al., 2021; Shi et al., 2020; Du et al., 2020; Qin et al., 2021a) as the base encoder for syntax inducing (pre-training) with dependency masking. For both BERT and XLNet, we try their base and large version following the default hyper-parameter settings, where their base version uses 12 layers of self-attentions with 768 dimensional hidden vectors and the large version uses 24 layers of self-attentions with 1024 dimensional hidden vectors for their large version.[11]

Pre-training Step	**700K**, 1,400K, 2,100K, 2,800K
Learning Rate	**1e-5**, 5e-5
Warmup Rate	**0.08**, 0.1, 0.2
Batch Size	**16**, 32

Table 2: The hyper-parameters tested in tuning our models for relation extraction. The best ones used in our final experiments are highlighted in boldface.

For syntax inducing, we train the model on the auto-parsed English Wiki for 700K steps[12] with the batch size set to 32. It is worth noting that, since English Wiki is used as a part of the data to train BERT and XLNet, it could be considered that we do not use additional data in experiments. For the process of fine-tuning the final RE model, we use the obtained syntax-induced encoder with randomly initialized bi-affine attentions. For other hyper-parameters, Table 2 reports the ones tested in training our models for training the relation extraction models. We test all combinations of them for each model and use the one achieving the highest results (i.e., F1 scores) on the development set. For evaluation, we follow previous studies to use the micro-F1 scores for ACE05 and use the official evaluation script[13] for SemEval.

4 Results and Analyses

4.1 Overall Results

Table 3 reports the results of our approach on the test set of ACE05 and SemEval with different encoders trained on first, second, and third order of dependencies (e.g., "+ DM (2nd)" denotes our approach with induced first and second order dependencies), as well as their corresponding baselines with only using the initial encoders (e.g., BERT and XLNet). We also run baselines with the standard graph convolutional networks (GCN) and the standard graph attentive networks (GAT) (Veličković et al., 2017) to leverage the auto-parsed dependency trees obtained in the same process as we obtain the auto-parsed Wiki (i.e., parsing and converting).

[6] https://catalog.ldc.upenn.edu/LDC99T42.

[7] We use the converter of version 3.3.0 from https://stanfordnlp.github.io/CoreNLP/index.html.

[8] We obtain the official data (LDC2006T06) from https://catalog.ldc.upenn.edu/LDC2006T06.

[9] The data is downloaded from http://docs.google.com/View?docid=dfvxd49s_36c28v9pmw.

[10] There is no official development set for ACE05.

[11] We download the cased version of BERT from https://github.com/google-research/bert and XLNet from https://github.com/zihangdai/xlnet.

[12] Syntax-induced encoder trained for 700K steps on the auto-parsed data achieves the optimal performance in most cases of the experiments (see more analyses in Section 4.4).

[13] We download the evaluation script from http://semeval2.fbk.eu/scorers/task08/SemEval2010_task8_scorer-v1.2.zip.

Models	ACE05	SemEval
BERT-Base	73.31	88.41
+ GCN	73.53	88.51
+ GAT	73.61	88.59
+ DM (1st)	73.62	88.65
+ DM (2nd)	**73.76**	88.60
+ DM (3rd)	73.65	**88.74**
BERT-Large	73.94	89.03
+ GCN	74.16	89.23
+ GAT	74.30	89.37
+ DM (1st)	74.34	89.42
+ DM (2nd)	**74.47**	**89.65**
+ DM (3rd)	74.29	89.37

(a) BERT-based Models

Models	ACE05	SemEval
XLNet-Base	73.42	88.78
+ GCN	73.55	88.84
+ GAT	73.67	88.90
+ DM (1st)	73.74	88.93
+ DM (2nd)	**73.86**	**89.11**
+ DM (3rd)	73.68	89.02
XLNet-Large	74.26	89.47
+ GCN	74.33	89.56
+ GAT	74.45	89.62
+ DM (1st)	74.41	89.60
+ DM (2nd)	**74.60**	**89.90**
+ DM (3rd)	74.51	89.76

(b) XLNet-based Models

Table 3: Experimental results of different models using base and large version of BERT and XLNet on the test set of ACE05 and SemEval. "+ GCN" and "+ GAT" refer to the models with the standard graph convolutional network and standard graph attentive networks, respectively. "+ DM" denotes our approaches with based encoder trained through dependency masking (DM) on word dependencies of different orders ("2nd" means both first and second order dependencies are masked and learnt, the same for "3rd").

There are several observations. First, our approach works well with different pre-trained language models (i.e., base and large BERT and XLNet), where the models with syntax-induced encoder outperform the vanilla BERT and XLNet baselines on both datasets, even though the baseline models have already achieved desirable performance. Second, compared with baseline models with standard GCN and GAT to leverage auto-parsed dependencies, our approach with different orders of dependency information consistently outperforms those baselines, which further confirms the effectiveness of our approach to leverage auto-parsed dependency information. Third, among models that leveraging dependency information in different orders, the ones with second order dependencies (i.e., "+ DM (2nd)") achieve the best performance in most cases. This observation confirms that RE models can benefit from high-order word dependencies since they provide association information among words with longer syntactic relations so as leading to better structure-aware understanding towards a sentence. However, it is still worth noting that, incorporating further higher order word dependencies (e.g., third order) may introduce noise or task-irrelevant information to the encoder since they are provided with auto-generated parses, which results in inferior performance comparing to using the second order dependencies.

4.2 Comparison with Previous Studies

We further compare our best performing model with previous studies on the test set of ACE05 and SemEval and report the results in Table 4. It is observed that, our approach outperforms all previous studies with different settings and encoders and achieves state-of-the-art scores on both datasets, which further confirms the effectiveness of our approach. Particularly, compared with previous studies (Zhang et al., 2018; Guo et al., 2019; Mandya et al., 2020; Sun et al., 2020; Yu et al., 2020b) that leverage the auto-parsed dependency tree of the input sentence through a particular module (e.g., Guo et al. (2019) proposed an graph-based approach with attentions to leverage dependency connections), where such dependency trees are required as extra input in inference, our approach uses an encoder to learn the dependency information through DMP and then fine-tune the obtained syntax-induced encoder on RE task. Such design in our approach allows our final RE model to be used without requiring the dependency tree of the sentence as the extra input in inference, which allows our model to run faster than previous approaches.

4.3 The Effect of Encoder Initialization

To explore the effect of encoder initialization with our approach, we run experiments by training our encoder starting from Transformer that uses the same architecture as BERT-base (i.e., 12 layers of

Models	ACE05	SemEval
Socher et al. (2012)	-	82.4
Zeng et al. (2014)	-	82.7
Zhang and Wang (2015)	-	79.6
Xu et al. (2015)	-	83.7
Wang et al. (2016)	-	88.0
Zhou et al. (2016)	-	84.0
†Zhang et al. (2018)	-	84.8
Wu and He (2019)	-	89.2
Christopoulou et al. (2018)	64.2	-
Ye et al. (2019)	68.9	-
†Guo et al. (2019)	-	85.4
Baldini Soares et al. (2019)	-	89.5
†Mandya et al. (2020)	-	85.9
†Sun et al. (2020)	-	86.0
†Yu et al. (2020a)	-	86.4
Wang et al. (2020)	66.7	-
Wang and Lu (2020)	67.6	-
Wang et al. (2021)	66.0	-
†Ours (BERT)	74.47	89.65
†Ours (XLNet)	**74.60**	**89.90**

Table 4: The comparison of F1 scores between previous studies and our best model with BERT-large on the test sets of ACE05 and SemEval. Previous studies that leverage syntactic information (e.g., the dependency tree of the input sentence) are marked by "†".

multi-head attentions with 768 dimensional hidden vectors) with random initialization (without using parameters from pre-trained language models or word embeddings). Table 5 reports the results of our approach when using different orders of dependency information, as well as the baseline results from the Transformer. As demonstrated, our approach significantly improves the baseline Transformer on both datasets, where around 30% absolute boost is observed on both ACE05 and SemEval datasets. This observation further confirms not only the effectiveness of our approach in improving base encoder with leveraging dependency information, but also its robustness of being applied to a randomly initialized base encoder.

4.4 The Effect of Training Steps

To analyze the performance change of the learned syntax-induced encoder on RE along with the increasing of training steps, we investigate the learned encoder (randomly initialized by a vanilla Transformer or pre-trained BERT-base model) with second order dependencies obtained from different training steps by fine-tuning it on ACE05 and SemEval. The test results (i.e., F1 scores) of our approach based on the vanilla Transformer and the BERT-base model with respect to the training steps (in 100 thousands) are illustrated in Figure 4 (a) and (b), respectively, and the performance of BERT-base baseline on different datasets is illustrated in dashed lines in different colors[14] in Figure 4 (b). In addition, we also evaluate the performance of the learned encoders (i.e., vanilla Transformer and the BERT-base model) trained by MDP on the test set of PTB for dependency parsing to illustrate how intensive of dependency information is introduced during the pre-training process[15], where the labeled attachment score (LAS) curves are presented in Figure 4 (c) for reference.

Models	ACE05	SemEval
Transformer	31.85	54.62
+ DM (1st)	**66.79**	79.37
+ DM (2nd)	66.67	**80.02**
+ DM (3rd)	64.54	79.95

Table 5: Comparisons of RE results from vanilla Transformer and our approach that being applied to a randomly initialized Transformer (without pre-trained language models or word embeddings).

It is shown that, when the Transformer is used, consistent improvements are observed with more training steps for both datasets. When a pre-trained language model (i.e., BERT-base) is used, it is observed that RE benefits much at the beginning of the pre-training (where the noisy auto-parsed dependency information is not intensively learning) and reach the peak (i.e., 74.11% for ACE05 and 89.02% for SemEval) when the training step reaches around 1,000K (where the syntax-induced encoder does not hurt by the noise in the dependencies). This phenomenon confirms the observations in previous studies (Xu et al., 2015; Zhang et al., 2018; Yu et al., 2020b; Sachan et al., 2021) that intensively leverage dependency information may introduce noise and confusion to relation classification, so that effective dependency pruning and introduce is of great importance. It also shows the effectiveness of our approach to address the noise by controlling the intensity of dependency information learning during pre-training.

[14]The performance on ACE05 and SemEval are illustrated in green and orange colors, respectively.

[15]Herein, the higher the performance of learned encoders on dependency parsing, the more intensive the dependency information is introduced in pre-training.

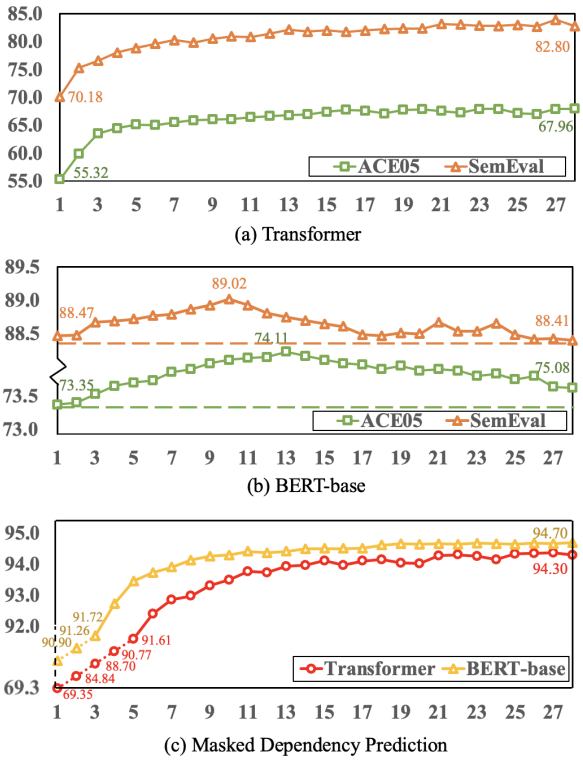

Figure 4: Curves of fine-tuning different base encoders (Transformer (a) and BERT-base (b)) on ACE05 and SemEval with respect to the number of training steps (in 100K). For reference, (c) shows the dependency parsing performance (LAS) of the learned encoders (Transformer and BERT-base) on the test set of English Penn Treebank (PTB) against its pre-training steps, where higher scores suggest that more intensive introduction of dependency information.

4.5 The Effect of Learned Representations

In previous results and analysis, we already show that the syntax-induced encoder outperforms baselines on RE with implicit integration of dependency information. Therefore, it is interesting to analyze the encoded word representations by qualitatively investigating their relations, which is similar to what has been done for word embeddings. In doing so, we collect word representations from the last layer of the trained syntax-induced encoder (XLNet-large). Then, for each word, we average its representation vectors under different contexts and use the resulting vector as its final representations. Figure 5 visualizes (by t-SNE) the representations of some example words, where the distance between two words indicates their similarity (closer distances indicate more relevant relations). It is observed that words with relevant syntactic properties (e.g., similar form or part-of-speech role) and semantic meanings are grouped into the same cluster (words in different clusters are

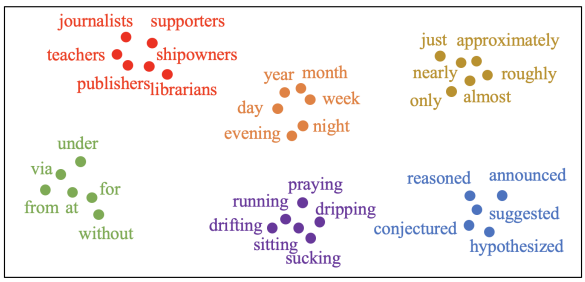

Figure 5: Visualizations of the learned representations through the syntax-induced encoder for some example words. The distance between any two words illustrates their similarity in terms of syntax and semantics. Words are presented in clusters and those in the same cluster are represented in the identical color.

represented in different colors). For example, all plural nouns of job names, e.g., "*teachers*", "*journalists*", "*publisher*", "*librarians*", "*shipowners*", and "*supporters*", are in the same cluster (represented in red color), while they are far away from irrelevant words, e.g., "*praying*". This finding is inspiring since such representations are automatically generated so that the MDP process shows its validity in learning syntax-aware word representations and ensuring that their relevance in syntax and semantics are appropriately modeled, which allows our model to achieve promising performance.

5 Related Work

Relation extraction is an important task in NLP and it requires deep understanding of the input text to achieve model performance. Therefore, in addition to leveraging advanced text encoders (e.g., bi-LSTM, Transformer (Vaswani et al., 2017), BERT (Devlin et al., 2019)) to capture contextual information, structural information, namely, the dependency information, of the running text has been widely used as an effective resource to improve RE (Xu et al., 2015; Zhang et al., 2018; Guo et al., 2019; Yu et al., 2020b; Chen et al., 2021). In most recent studies in NLP, the dependency information is leveraged either as extra input features (Prokopidis and Papageorgiou, 2014; Kiperwasser and Goldberg, 2015; Yu and Bohnet, 2017) or modeled by complicated graph-based architectures, such as convolutions neural networks (Marcheggiani and Titov, 2017; Zhang et al., 2018) and tree LSTMs (Peng et al., 2017; Li et al., 2018). Previous studies also tried to use attention mechanism to weight different dependency features (Guo et al., 2019; Yu et al., 2020b; Qin et al., 2021b) and LSTM to en-

code linearized dependency path (Xu et al., 2015; Roth and Lapata, 2016). In addition to modeling dependency information, there is another track to leverage it by pre-training dependency-based word embeddings through predicting the context words in auto-parsed dependency trees (Levy and Goldberg, 2014; Komninos and Manandhar, 2016) or designing an auxiliary module to learn the dependency information by treating the dependencies as additional input during pre-training (Xu et al., 2021). This research follows the pre-training paradigm and offers an alternative way to do so.

Specifically, compared with existing studies, our approach leverages the dependency information by inducing it to the pre-training process through masked dependency prediction, whose object is to predict the masked dependencies rather than directly using it as extra fixed input along with the input sentence through an additional module. Also, since the dependency information is learnt by the syntax-induced encoder and the encoder is further fine-tuned on the training data in the same way as general RE model, our approach neither requires any additional input features nor needs complicated architectures to encode them, which allows our model to be efficient in inference.

6 Conclusion

In this paper, we propose to use dependency masking and recovering to improve the text encoder and thus enhance RE that requires deep understanding of the running text, where the encoder is trained on large scaled auto-parsed data. Specifically, we try such masking on first, second, and third order word dependencies from the auto-parsed data, and train a base encoder that is able to recover all the masked dependencies. In doing so, the resulted syntax-induced encoder is integrated with dependency information in a dynamic and flexible manner and it can be directly applied to different downstream tasks requiring no extra input or particular design to accommodate dependency information. Experimental results and analyses on two English benchmark datasets (i.e., ACE05 and SemEval) for RE show the effectiveness of our approach, where our approach outperforms strong baselines and achieve state-of-the-art on both datasets.

Acknowledgements

This work is supported by Shenzhen Science and Technology Program under the project "Fundamental Algorithms of Natural Language Understanding for Chinese Medical Text Processing" (JCYJ20210324130208022) and the Natural Science Foundation of Guangdong Province, China, under the project "Deep Learning based Chinese Combination Category Grammar Parsing and its Application in Relation Extraction". It is also supported by Shenzhen Institute of Artificial Intelligence and Robotics for Society under the project "Automatic Knowledge Enhanced Natural Language Understanding and Its Applications" (AC01202101001).

References

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

Guimin Chen, Yuanhe Tian, Yan Song, and Xiang Wan. 2021. Relation Extraction with Type-aware Map Memories of Word Dependencies. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2501–2512, Online.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. A Walk-based Model on Entity Graphs for Relation Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 81–88.

Greg Coppola and Mark Steedman. 2013. The Effect of Higher-order Dependency Features in Discriminative Phrase-structure Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 610–616, Sofia, Bulgaria.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bayu Distiawan, Gerhard Weikum, Jianzhong Qi, and Rui Zhang. 2019. Neural Relation Extraction for Knowledge Base Enrichment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 229–240.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware BERT for cross-domain sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4019–4028, Online.

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. 2010. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38.

Tao Ji, Yuanbin Wu, and Man Lan. 2019. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485, Florence, Italy.

Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. 2018. Higher-order Syntactic Attention Network for Longer Sentence Compression. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1716–1726, New Orleans, Louisiana.

Zhen Ke, Liang Shi, Songtao Sun, Erli Meng, Bin Wang, and Xipeng Qiu. 2021. Pre-training with Meta Learning for Chinese Word Segmentation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5514–5523, Online.

Eliyahu Kiperwasser and Yoav Goldberg. 2015. Semi-supervised Dependency Parsing using Bilexical Contextual Features from Auto-Parsed Data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1348–1353, Lisbon, Portugal.

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)*, pages 2676–2686, Melbourne, Australia.

Alexandros Komninos and Suresh Manandhar. 2016. Dependency Based Embeddings for Sentence Classification Tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1490–1500, San Diego, California.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020a. Structured Tuning for Semantic Role Labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online.

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. 2018. A Unified Syntax-aware Framework for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411, Brussels, Belgium.

Zuchao Li, Hai Zhao, Rui Wang, and Kevin Parnow. 2020b. High-order Semantic Role Labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1134–1151, Online.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Angrosh Mandya, Danushka Bollegala, and Frans Coenen. 2020. Graph Convolution over Multiple Dependency Sub-graphs for Relation Extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6424–6435.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

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*, pages 1506–1515, Copenhagen, Denmark.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245.

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.

Prokopis Prokopidis and Haris Papageorgiou. 2014. Experiments for Dependency Parsing of Greek. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 90–96, Dublin, Ireland.

Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. 2021a. Improving Arabic Diacritization with Regularized Decoding and Adversarial Training. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Han Qin, Yuanhe Tian, and Yan Song. 2021b. Relation Extraction with Word Graphs from N-grams. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2860–2868, Online and Punta Cana, Dominican Republic.

Michael Roth and Mirella Lapata. 2016. Neural Semantic Role Labeling with Dependency Path Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany.

Devendra Sachan, Yuhao Zhang, Peng Qi, and William L. Hamilton. 2021. Do Syntax Trees Help Pre-trained Transformers Extract Information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online.

Tianze Shi, Igor Malioutov, and Ozan Irsoy. 2020. Semantic Role Labeling as Syntactic Dependency Parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7551–7571, Online.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211.

Yan Song and Shuming Shi. 2018. Complementary Learning of Word Embeddings. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4368–4374.

Yan Song, Tong Zhang, Yonggang Wang, and Kai-Fu Lee. 2021. ZEN 2.0: Continue Training and Adaption for N-gram Enhanced Text Encoders. *arXiv preprint arXiv:2105.01279*.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation Extraction with Convolutional Network over Learnable Syntax-Transport Graph. In *AAAI*, pages 8928–8935.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. *arXiv preprint arXiv:2107.02137*.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021. Dependency-driven Relation Extraction with Attentive Graph Convolutional Networks. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020. Improving Constituency Parsing with Span Attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph Attention Networks. *arXiv preprint arXiv:1710.10903*.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online.

Linlin Wang, Zhu Cao, Gerard De Melo, and Zhiyuan Liu. 2016. Relation Classification via Multi-Level Attention CNNs. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1298–1307.

Lu Wang and Claire Cardie. 2012. Focused Meeting Summarization via Unsupervised Relation Extraction. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 304–313.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Junchi Yan, Peng Gao, and Guotong Xie. 2020. Pre-training entity relation encoder with intra-span and inter-span information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1692–1705, Online.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021. UNIRE: A Unified Label Space for Entity Relation Extraction. *arXiv preprint arXiv:2107.04292*.

Shanchan Wu and Yifan He. 2019. Enriching Pretrained Language Model with Entity Information for Relation Classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2361–2364.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question Answering on Freebase via Relation Extraction and Textual Evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2326–2336.

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 *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-Enhanced Pre-trained Model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online.

Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. A Graph-based Model for Joint Chinese Word Segmentation and Dependency Parsing. *Transactions of the Association for Computational Linguistics*, 8:78–92.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems 32*, pages 5753–5763.

Wei Ye, Bo Li, Rui Xie, Zhonghao Sheng, Long Chen, and Shikun Zhang. 2019. Exploiting Entity BIO Tag Embeddings and Multi-task Learning for Relation Extraction with Imbalanced Data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1351–1360.

Bowen Yu, Mengge Xue, Zhenyu Zhang, Tingwen Liu, Yubin Wang, and Bin Wang. 2020a. Learning to Prune Dependency Trees with Rethinking for Neural Relation Extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3842–3852.

Bowen Yu, Mengge Xue, Zhenyu Zhang, Tingwen Liu, Wang Yubin, and Bin Wang. 2020b. Learning to Prune Dependency Trees with Rethinking for Neural Relation Extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3842–3852, Barcelona, Spain (Online).

Juntao Yu and Bernd Bohnet. 2017. Dependency language models for transition-based dependency parsing. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 11–17, Pisa, Italy.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation Classification via Convolutional Deep Neural Network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45.

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 *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.