Syntax-Enhanced Pre-trained Model

Zenan Xu[1*†], **Daya Guo**[1*], **Duyu Tang**[2†], **Qinliang Su**[1,4,5‡], **Linjun Shou**[3], **Ming Gong**[3], **Wanjun Zhong**[1*], **Xiaojun Quan**[1], **Daxin Jiang**[3], **and Nan Duan**[2]
[1]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
[2]Microsoft Research Asia, Beijing, China
[3]Microsoft Search Technology Center Asia, Beijing, China
[4]Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
[5]Key Lab. of Machine Intelligence and Advanced Computing, Ministry of Education, China
{xuzn, guody5, zhongwj25}@mail2.sysu.edu.cn
{suqliang, quanxj3}@mail.sysu.edu.cn
{dutang,lisho,migon,djiang,nanduan}@microsoft.com

Abstract

We study the problem of leveraging the syntactic structure of text to enhance pre-trained models such as BERT and RoBERTa. Existing methods utilize syntax of text either in the pre-training stage or in the fine-tuning stage, so that they suffer from discrepancy between the two stages. Such a problem would lead to the necessity of having human-annotated syntactic information, which limits the application of existing methods to broader scenarios. To address this, we present a model that utilizes the syntax of text in both pre-training and fine-tuning stages. Our model is based on Transformer with a syntax-aware attention layer that considers the dependency tree of the text. We further introduce a new pre-training task of predicting the syntactic distance among tokens in the dependency tree. We evaluate the model on three downstream tasks, including relation classification, entity typing, and question answering. Results show that our model achieves state-of-the-art performance on six public benchmark datasets. We have two major findings. First, we demonstrate that infusing automatically produced syntax of text improves pre-trained models. Second, *global* syntactic distances among tokens bring larger performance gains compared to *local* head relations between contiguous tokens.[1]

1 Introduction

Pre-trained models such as BERT (Devlin et al., 2019), GPT (Radford et al., 2018), and RoBERTa (Liu et al., 2019) have advanced the state-of-the-art performances of various natural language processing tasks. The successful recipe is that a model is first pre-trained on a huge volume of unsupervised data with self-supervised objectives, and then is fine-tuned on supervised data with the same data scheme. Dominant pre-trained models represent a text as a sequence of tokens[2]. The merits are that such basic text representations are available from vast amounts of unsupervised data, and that models pre-trained and fine-tuned with the same paradigm usually achieve good accuracy in practice (Guu et al., 2020). However, an evident limitation of these methods is that richer syntactic structure of text is ignored.

In this paper, we seek to enhance pre-trained models with syntax of text. Related studies attempt to inject syntax information either only in the fine-tuning stage (Nguyen et al., 2020; Sachan et al., 2020), or only in the pre-training stage (Wang et al., 2020), which results in discrepancies. When only fusing syntax information in the fine-tuning phase, Sachan et al. (2020) finds that there is no performance boost unless high quality human-annotated dependency parses are available. However, this requirement would limit the application of the model to broader scenarios where human-annotated dependency information is not available.

To address this, we conduct a large-scale study on injecting automatically produced syntax of text in both the pre-training and fine-tuning stages. We construct a pre-training dataset by applying an off-the-shelf dependency parser (Qi et al., 2020) to one billion sentences from common crawl news. With these data, we introduce a syntax-aware pre-training task, called dependency distance prediction, which predicts the syntactic distance between tokens in the dependency structure. Compared with the pre-training task of dependency head prediction (Wang et al., 2020) that only captures local syntactic relations among words, dependency distance prediction leverages global syntax of the text. In

[*] Work is done during internship at Microsoft.
[†] For questions, please contact D. Tang and Z. Xu.
[‡] Corresponding author.
[1]The source data is available at https://github.com/Hi-ZenanXu/Syntax-Enhanced_Pre-trained_Model.
[2]Such tokens can be words or word pieces. We use token for clarity.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 5412–5422
August 1–6, 2021.

addition, we developed a syntax-aware attention layer, which can be conveniently integrated into Transformer (Vaswani et al., 2017) to allow tokens to selectively attend to contextual tokens based on their syntactic distance in the dependency structure.

We conduct experiments on entity typing, question answering and relation classification on six benchmark datasets. Experimental results show that our method achieves state-of-the-art performance on all six datasets. Further analysis shows that our model can indicate the importance of syntactic information on downstream tasks, and that the newly introduced dependency distance prediction task could capture the global syntax of the text, performs better than dependency head prediction. In addition, compared with experimental results of injecting syntax information in either the pre-training or fine-tuning stage, injecting syntax information in both stages achieves the best performance.

In summary, the contribution of this paper is threefold. (1) We demonstrate that infusing automatically produced dependency structures into the pre-trained model shows superior performance over downstream tasks. (2) We propose a syntax-aware attention layer and a pre-training task for infusing syntactic information into the pre-trained model. (3) We find that the newly introduced dependency distance prediction task performs better than the dependency head prediction task.

2 Related Work

Our work involves injecting syntax information into pre-trained models. First, we will review recent studies on analyzing the knowledge presented in pre-trained models, and then we will introduce the existing methods that enhance pre-trained models with syntax information.

2.1 Probing Pre-trained Models

With the huge success of pre-trained models (Devlin et al., 2019; Radford et al., 2018) in a wide range of NLP tasks, lots of works study to what extent pre-trained models inherently. Here, we will introduce recent works on probing linguistic information, factual knowledge, and symbolic reasoning ability from pre-trained models respectively. In terms of linguistic information, Hewitt and Manning (2019) learn a linear transformation to predict the depth of each word on a syntax tree based on their representation, which indicates that the syntax information is implicitly embedded in the BERT model. However, Yaushian et al. (2019) find that the attention scores calculated by pre-trained models seem to be inconsistent with human intuitions of hierarchical structures, and indicate that certain complex syntax information may not be naturally embedded in BERT. In terms of probing factual knowledge, Petroni et al. (2019) find that pre-trained models are able to answer fact-filling cloze tests, which indicates that the pre-trained models have memorized factual knowledge. However, Poerner et al. (2019) argue that BERT's outstanding performance of answering fact-filling cloze tests is partly due to the reasoning of the surface form of name patterns. In terms of symbolic reasoning, Talmor et al. (2020) test the pre-trained models on eight reasoning tasks and find that the models completely fail on half of the tasks. Although probing knowledge from pre-trained model is a worthwhile area, it runs perpendicular to infusing knowledge into pre-trained models.

2.2 Integrating Syntax into Pre-trained Models

Recently, there has been growing interest in enhancing pre-trained models with syntax of text. Existing methods attempt to inject syntax information in the fine-tuning stage or only in the pre-training stage. We first introduce related works that inject syntax in the fine-tuning stage. Nguyen et al. (2020) incorporate a tree-structured attention into the Transformer framework to help encode syntax information in the fine-tuning stage. Zhang et al. (2020) utilize the syntax to guide the Transformer model to pay no attention to the dispensable words in the fine-tuning stage and improve the performance in machine reading comprehension. Sachan et al. (2020) investigate two distinct strategies for incorporating dependency structures in the fine-tuning stage and obtain state-of-the-art results on the semantic role labeling task. Meanwhile, Sachan et al. (2020) argue that the performance boost is mainly contributed to the high-quality human-annotated syntax. However, human annotation is costly and difficult to extend to a wide range of applications. Syntax information can also be injected in the pre-training stage. Wang et al. (2020) introduce head prediction tasks to inject syntax information into the pre-trained model, while syntax information is not provided during inference. Note that the head prediction task in Wang et al. (2020) only focuses

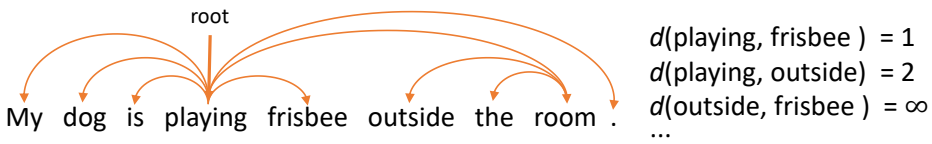


Figure 1: The dependency tree of the sentence, "My dog is playing frisbee outside the room," after running the Stanza parser.

on the local relationship between two related tokens, which prevents each token from being able to perceive the information of the entire tree. Despite the success of utilizing syntax information, existing methods only consider the syntactic information of text in the pre-training or the fine-tuning stage so that they suffer from discrepancy between the pre-training and the fine-tuning stage. To bridge this gap, we conduct a large-scale study on injecting automatically produced syntax information in both the two stages. Compared with the head prediction task (Wang et al., 2020) that captures the local relationship, we introduce the dependency distance prediction task that leverages the global relationship to predict the distance of two given tokens.

3 Data Construction

In this paper, we adopt the dependency tree to express the syntax information. Such a tree structure is concise and only expresses necessary information for the parse (Jurafsky, 2000). Meanwhile, its head-dependent relation can be viewed as an approximation to the semantic relationship between tokens, which is directly useful for capturing semantic information. The above advantages help our model make more effective use of syntax information. Another available type of syntax information is the constituency tree, which is used in Nguyen et al. (2020). However, as pointed out in Jurafsky (2000), the relationships between the tokens in dependency tree can directly reflect important syntax information, which is often buried in the more complex constituency trees. This property requires extra techniques to extracting relation among the words from a constituency tree (Jurafsky, 2000)[3].

The dependency tree takes linguistic words as one of its basic units. However, most pre-trained models take subwords (also known as the word pieces) instead of the entire linguistic words as the input unit, and this necessitates us to extend the definition of the dependency tree to include subwords. Following Wang et al. (2020), we will add edges from the first subword of v to all subwords of u, if there exists a relationship between linguistic word v and word u.

Based on the above extended definition, we build a pre-training dataset from open-domain sources. Specifically, we randomly collect 1B sentences from publicly released common crawl news datasets (Zellers et al., 2019) that contain English news articles crawled between December 2016 and March 2019. Considering its effectiveness and ability to expand to multiple languages, we adopt off-the-shelf Stanza[4] to automatically generate the syntax information for each sentence. The average token length of each sentence is 25.34, and the average depth of syntax trees is 5.15.

4 Methodology

In this section, we present the proposed **S**yntax-**E**nhanced **PRE**-trained **M**odel (**SEPREM**). We first define the syntax distance between two tokens. Based on the syntax distance, we then introduce a syntax-aware attention layer to learn syntax-aware representation and a pre-training task to enable model to capture global syntactic relations among tokens.

4.1 Syntax Distance over Syntactic Tree

Intuitively, the distance between two tokens on the syntactic tree may reflect the strength of their linguistic correlation. If two tokens are far away from each other on the syntactic tree, the strength of their linguistic correlation is likely weak. Thus, we define the distance of two tokens over the dependency tree as their syntactic distance. Specifically, we define the distance between the token v and token u as 1, i.e. $d(v, u) = 1$, if v is the head of u. If two tokens are not directly connected in the dependency graph, their distance is the summation of the distances between adjacent nodes on the path. If two tokens are separated in the graph, their distance is set to infinite. Taking the sentence "*My dog is playing frisbee outside the room.*" in Fig 1 as

[3] https://web.stanford.edu/~jurafsky/slp3/

[4] https://github.com/stanfordnlp/stanza

an example, $d(\textit{playing}, \textit{frisbee})$ equals 1 since the token "*playing*" is the head of the token "*frisbee*".

4.2 Syntax-Aware Transformer

We follow BERT (Devlin et al., 2019) and use the multi-layer bidirectional Transformer (Vaswani et al., 2017) as the model backbone. The model takes a sequence X as the input and applies N transformer layers to produce contextual representation:

$$\boldsymbol{H}^n = transformer_n((1-\alpha)\boldsymbol{H}^{n-1} + \alpha\hat{\boldsymbol{H}}^{n-1}) \tag{1}$$

where $n \in [1, N]$ denotes the n-th layer of the model, $\hat{\boldsymbol{H}}$ is the syntax-aware representation which will be described in Section 4.3, $\boldsymbol{H}^0$ is embeddings of the sequence input X, and α is a learnable variable.

However, the introduction of syntax-aware representation $\hat{\boldsymbol{H}}$ in the Equation 1 changes the architecture of Transformer, invalidating the original weights from pre-trained model, such as BERT and RoBERTa. Instead, we introduce a learnable importance score α that controls the proportion of integration between contextual and syntax-aware representation. When α is equal to zero, the syntax-aware representation is totally excluded and the model is architectural identical to vanilla Transformer. Therefore, we initialize the parameter α as the small but not zero value, which can help better fuse syntactic information into existing pre-trained models. We will discuss importance score α in detailed in Section 5.6.

Each transformer layer $transformer_n$ contains an architecturally identical transformer block, which is composed of a multi-headed self-attention $MultiAttn$ (Vaswani et al., 2017) and a followed feed forward layer FFN. Formally, the output $\boldsymbol{H}^n$ of the transformer block $transformer_n(H'_{n-1})$ is computed as:

$$\begin{aligned} \boldsymbol{G}'_n &= LN(MultiAttn(H'_{n-1}) + H'_{n-1}) \\ \boldsymbol{H}^n &= LN(FFN(\boldsymbol{G}'_n) + \boldsymbol{G}'_n) \end{aligned} \tag{2}$$

where the input H'_{n-1} is $(1-\alpha)\boldsymbol{H}^{n-1} + \alpha\hat{\boldsymbol{H}}^{n-1}$ and LN represents a layer normalization operation.

4.3 Syntax-aware Attention Layer

In this section, we will introduce how to obtain the syntax-aware representation $\hat{\boldsymbol{H}}$ used in syntax-aware transformer.

Tree Structure Encoding We adopt a distance matrix $\boldsymbol{D}$ to encode the tree structure. The advantages of distance matrix $\boldsymbol{D}$ are that it can well preserve the hierarchical syntactic structure of text and can directly reflect the distance of two given tokens. Meanwhile, its uniqueness property guarantees the one-to-one mapping of the tree structure. Given a dependency tree, the element $\boldsymbol{D}_{i,j}$ of distance matrix $\boldsymbol{D}$ in i-th row and j-th column is defined as:

$$\boldsymbol{D}_{i,j} = \begin{cases} d(i,j), & \text{if exists a path from } v_i \text{ to } v_j, \\ 0, & \text{if } i = j \text{ and otherwise.} \end{cases} \tag{3}$$

where v_i and v_j are tokens on the dependency tree. Based on the concept that distance is inversely proportional to importance, we normalize the matrix $\boldsymbol{D}$ and obtain the normalized correlation strength matrix $\tilde{\boldsymbol{D}}$ as follows:

$$\tilde{\boldsymbol{D}}_{i,j} = \begin{cases} \frac{1/\boldsymbol{D}_{i,j}}{\sum_{z \in \{y | \boldsymbol{D}_{i,y} \neq 0\}} (1/\boldsymbol{D}_{i,z})}, & \text{if } \boldsymbol{D}_{i,j} \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Syntax-aware Representation Given the tree structure representation $\tilde{\boldsymbol{D}}$ and the contextual representation $\boldsymbol{H}^n$, we fuse the tree structure into the contextual representation as:

$$\hat{\boldsymbol{H}}^n = \sigma(\boldsymbol{W}_n^1 \boldsymbol{H}^n + \boldsymbol{W}_n^2 \tilde{\boldsymbol{D}} \boldsymbol{H}^n) \tag{5}$$

where σ is the activation function, $\boldsymbol{W}_n^1$ and $\boldsymbol{W}_n^2 \in \mathbb{R}^{d_h \times d_h}$ are model parameters. We can see that $\tilde{\boldsymbol{D}}\boldsymbol{H}^n$ allows one to aggregate information from others along the tree structure. The closer they are on the dependency tree, the larger the attention weight, and thus more information will be propagated to each other, and vice verse.

4.4 Syntax-aware Pre-training Task

To better understand the sentences, it is beneficial for model to be aware of the underlying syntax. To this end, a new pre-training task, named dependency distance prediction task (DP), is designed to enhance the model's ability of capturing global syntactic relations among tokens. Specifically, we first randomly mask some elements in the distance matrix $\boldsymbol{D}$, e.g., supposed $\boldsymbol{D}_{i,j}$. Afterwards, the representations of tokens i and j from SEPREM are concatenated and fed into a linear classifier, which outputs the probabilities over difference distances. In all of our experiments, 15% of distance are masked at random.

Similar to BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), we conduct the following operations to boost the robustness. The distance in matrix $\boldsymbol{D}$ will be masked at 80% probability or replaced by a random integer with a probability of 10%. For the rest 10% probability, the distance will be maintained.

During pre-training, in addition to the DP pre-training task, we also use the dependency head prediction (HP) task, which is used in Wang et al. (2020) to capture the local head relation among words, and the dynamic masked language model (MLM), which is used in Liu et al. (2019) to capture contextual information. The final loss for the pre-training is the summation of the training loss of DP, HP and MLM tasks.

4.5 Implementation Details

The implementation of SEPREM is based on HuggingFace's Transformer (Wolf et al., 2019). To accelerate the training process, we initialize parameters from RoBERTa model released by HuggingFace[5], which contains 24 layers, with 1024 hidden states in each layer. The number of parameters of our model is 464M. We pre-train our model with 16 32G NVIDIA V100 GPUs for approximately two weeks. The batch size is set to 2048, and the total steps are 500000, of which 30000 is the warm up steps.

In both pre-training and fine-tuning stages, our model takes the syntax of the text as the additional input, which is pre-processed in advance. Specially, we obtain the dependency tree of each sentence via Stanza and then generate the normalized distance matrix.

5 Experiments

In this section, we evaluate the proposed SEPREM on six benchmark datasets over three downstream tasks, *i.e.*, entity typing, question answering and relation classification.

5.1 Entity Typing

The entity typing task requires the model to predict the type of a given entity based on its context. Two fine-grained public datasets, Open Entity (Choi et al., 2018) and FIGER (Ling et al., 2015), are employed to evaluate our model. The statistics of the aforementioned datasets are shown in Table 1. Following Wang et al. (2020), special token "@" is added before and after a certain entity, then the representation of the first special token "@" is adopted to predict the type of the given entity. To keep the evaluation criteria consistent with previous works (Shimaoka et al., 2016; Zhang et al., 2019; Peters et al., 2019; Wang et al., 2019; Xiong et al., 2020), we adopt loose micro precision, recall, and F1 to evaluate model performance on Open Entity datasets. As for FIGER datasets, we utilize strict accuracy, loose macro-F1, and loose micro-F1 as evaluation metrics.

Dataset	Train	Dev	Test	Label
Open Entity	2,000	2,000	2,000	6
FIGER	2,000,000	10,000	563	113
TACRED	68,124	22,631	15,509	42

Table 1: The statistics of the entity typing datasets, i.e., Open Entity and FIGER, and relation classification dataset TACRED. Label refers to type of a given entity or relation between two entities.

Baselines NFGEC (Shimaoka et al., 2016) recursively composes representation of entity context and further incorporates an attention mechanism to capture fine-grained category memberships of an entity. KEPLER (Wang et al., 2019) infuses knowledge into the pre-trained models and jointly learns the knowledge embeddings and language representation. RoBERTa-large (continue training) learns on the proposed pre-training dataset under the same settings with SEPREM but only with dynamic MLM task. In addition, we also report the results of BERT-base (Devlin et al., 2019), ERNIE (Zhang et al., 2019), KnowBERT (Peters et al., 2019), WKLM (Xiong et al., 2020), RoBERTa-large, and K-adapter (Wang et al., 2020) for a full comparison.

Experimental Results As we can see in Table 2, our SEPREM outperforms all other baselines on both entity typing datasets. In the Open Entity dataset, with the utility of the syntax of text, SEPREM achieves an improvement of 3.6% in micro-F1 score comparing with RoBERTa-large (continue training) model. The result demonstrates that the proposed syntax-aware pre-training tasks and syntax-aware attention layer help to capture the syntax of text, which is beneficial to predict the types more accurately. As for the FIGER dataset, which contains more labels about the type of entity, SEPREM still brings an improvement in strict accuracy, macro-F1, and micro-F1. This demonstrates

[5] https://huggingface.co/transformers/

Model	OpenEntity			FIGER		
	P	R	Mi-F$_1$	Acc	Ma-F$_1$	Mi-F$_1$
NFGEC (Shimaoka et al., 2016)	68.80	53.30	60.10	55.60	75.15	71.73
BERT-base (Zhang et al., 2019)	76.37	70.96	73.56	52.04	75.16	71.63
ERNIE (Zhang et al., 2019)	78.42	72.90	75.56	57.19	75.61	73.39
KnowBERT (Peters et al., 2019)	78.60	73.70	76.10	-	-	-
KEPLER (Wang et al., 2019)	77.20	74.20	75.70	-	-	-
WKLM (Xiong et al., 2020)	-	-	-	60.21	81.99	77.00
K-Adapter (Wang et al., 2020)	79.25	75.00	77.06	61.81	84.87	80.54
RoBERTa-large	77.55	74.95	76.23	56.31	82.43	77.83
RoBERTa-large (continue training)	77.63	75.01	76.30	56.52	82.37	77.81
SEPREM	**81.07**	**77.14**	**79.06**	**63.21**	**86.14**	**82.05**

Table 2: Results for entity typing task on the OpenEntity and FIGER datasets.

Dataset	Train	Dev	Test
SearchQA	99,811	13,893	27,247
Quasar-T	28,496	3,000	3,000
CosmosQA	25,588	3,000	7,000

Table 3: The statistics of the question answering datasets: SearchQA, Quasar-T and CosmosQA.

the effectiveness of leveraging syntactic information in tasks with more fine-grained information. Specifically, compared with the K-adapter model, our SEPREM model brings an improvement of 2.6% F1 score on Open Entity dataset. It is worth noting that SEPREM model is complementary to the K-adapter model, both of which inject syntactic information into model during pre-training stage. This improvement indicates that injecting syntactic information in both the pre-training and fine-tuning stages can make full use of the syntax of the text, thereby benefiting downstream tasks.

5.2 Question Answering

We use open-domain question answering (QA) task and commonsense QA task to evaluate the proposed model. Open-domain QA requires models to answer open-domain questions with the help of external resources such as materials of collected documents and webpages. We use SearchQA (Dunn et al., 2017) and QuasarT (Dhingra et al., 2017) for this task, and adopt ExactMatch (EM) and loose F1 scores as evaluation metrics. In this task, we first retrieve related paragraphs according to the question from external materials via the information retrieval system, and then a reading comprehension technique is adopted to extract possible answers from the above retrieved paragraphs. Following previous work (Lin et al., 2018), we use the retrieved paragraphs provided by Wang et al. (2017b) for the two datasets. For fair comparison, we follow Wang et al. (2020) to use $[<sep>, quesiton, </sep>, paragraph, </sep>]$ as the input, where $<sep>$ is a special token in front of two segmants and $</sep>$ is a special symbol to split two kinds of data types. We take the task as a multi-classification to fine-tune the model and use two linear layers over the last hidden features from models to predict the start and end positions of the answer span.

Commonsense QA aims to answer questions which require commonsense knowledge that is not explicitly expressed in the question. We use the public CosmosQA dataset (Huang et al., 2019) for this task, and the accuracy scores are used as evaluation metrics. The data statistics of the above three datasets are shown in Table 3. In CosmosQA, each question has 4 candidate answers, and we concatenate the question together with each answer separately as $[<sep>, context, </sep>, paragraph, </sep>]$ for input. The representation of the first token is adopted to calculate a score for this answer, and the answer with the highest score is regarded as the prediction answer for this question.

Baselines BiDAF (Seo et al., 2017) is a bidirectional attention network to obtain query-aware context representation. AQA (Buck et al., 2018) adopts a reinforce-guide questions re-write system and generates answers according to the re-written questions. Rˆ3 (Wang et al., 2017a) selects the most

Model	SearchQA		Quasar-T		CosmosQA
	EM	F_1	EM	F_1	Accuracy
BiDAF (Seo et al., 2017)	28.60	34.60	25.90	28.50	-
AQA (Buck et al., 2018)	40.50	47.40	-	-	-
Rˆ3 (Wang et al., 2017a)	49.00	55.30	35.30	41.70	-
DSQA (Lin et al., 2018)	49.00	55.30	42.30	49.30	-
Evidence Agg. (Wang et al., 2018)	57.00	63.20	42.30	49.60	-
BERT (Xiong et al., 2020)	57.10	61.90	40.40	46.10	-
WKLM (Xiong et al., 2020)	58.70	63.30	43.70	49.90	-
WKLM + Ranking (Xiong et al., 2020)	61.70	66.70	45.80	52.20	-
BERT-FT$_{RACE+SWAG}$ (Huang et al., 2019)	-	-	-	-	68.70
K-ADAPTER (Wang et al., 2020)	61.96	67.31	45.69	52.48	81.83
RoBERTa-large	59.01	65.62	40.83	48.84	80.59
RoBERTa-large (continue training)	59.34	65.71	40.91	49.04	80.75
SEPREM	**62.31**	**67.74**	**46.37**	**53.18**	**82.37**

Table 4: Results on QA datasets including: SearchQA, Quasar-T and CosmosQA.

Model	P	R	F_1
C-GCN (Zhang et al., 2018)	69.90	63.30	66.40
BERT-base (Zhang et al., 2019)	67.23	64.81	66.00
ERNIE (Zhang et al., 2019)	69.97	66.08	67.97
BERT-large (Baldini Soares et al., 2019)	-	-	70.10
BERT+MTB (Baldini Soares et al., 2019)	-	-	71.50
KnowBERT (Peters et al., 2019)	71.60	71.40	71.50
KEPLER (Wang et al., 2019)	70.43	73.02	71.70
K-Adapter (Wang et al., 2020)	70.05	73.92	71.93
RoBERTa-large	70.17	72.36	71.25
RoBERTa-large (continue training)	70.19	72.41	71.28
SEPREM	**70.57**	**74.36**	**72.42**

Table 5: Results for relation classification task on TACRED dataset.

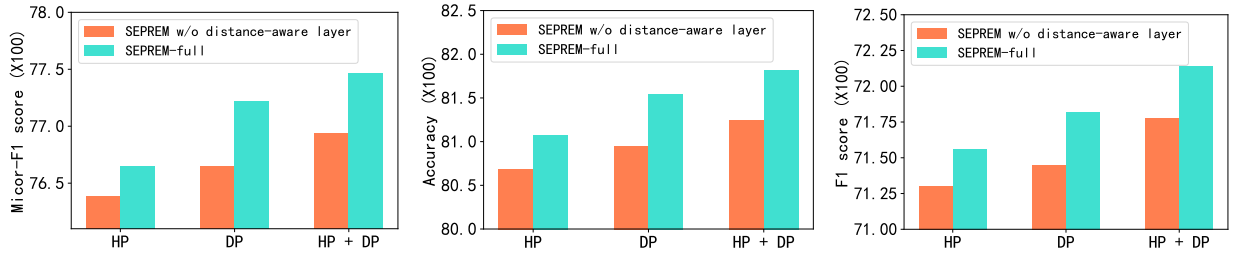
confident paragraph with a designed reinforcement ranker. DSQA (Lin et al., 2018) employs a paragraph selector to remove paragraphs with noise and a paragraph reader to extract the correct answer from denoised paragraphs. Evidence Agg. (Wang et al., 2018) makes use of multiple passages to generate answers. BERT-FT$_{RACE+SWAG}$ (Huang et al., 2019) sequentially fine-tunes the BERT model on the RACE and SWAG datasets for knowledge transfer. Besides the aforementioned models, we also report the results of BERT (Xiong et al., 2020), WKLM (Xiong et al., 2020), WKLM + Ranking (Xiong et al., 2020), RoBERTa-large, RoBERTa-large (continue training), and K-Adapter (Wang et al., 2020) for a detailed comparison.

Experimental Results The results of the open-domain QA task are shown in Table 4. We can see that the proposed SEPREM model brings significant gains of 3.1% and 8.4% in F1 scores, compared with RoBERTa-large (continue training) model. This may be partially attributed to the fact that, QA task requires a model to have reading comprehension ability (Wang et al., 2020), and the introduced syntax information can guide the model to avoid concentrating on certain dispensable words and improve its reading comprehension capacity (Zhang et al., 2020). Meanwhile, SEPREM achieves state-of-the-art results on the CosmosQA dataset, which demonstrates the effectiveness of the proposed SEPREM model. It can be also seen that the performance gains observed in CosmosQA are not as substantial as those in the open-domain QA tasks. We speculate that CosmosQA requires capacity for contextual commonsense reasoning and the lack of explicitly injection of commonsense knowledge into SEPREM model limits its improvement.

5.3 Relation Classification

A relation classification task aims to predict the relation between two given entities in a sentence. We use a large-scale relation classification dataset TACRED (Zhang et al., 2017) for this task, and adopt Micro-precision, recall, and F1 scores as evaluation metrics. The statistics of the TACRED datasets are shown in Table 1. Following Wang et al. (2020), we add special tokens "@" and "#" before and after the first and second entity respectively. Then, the representations of the former token "@" and "#" are concatenated to perform relation classification.

Baselines C-GCN (Zhang et al., 2018) encodes the dependency tree via graph convolutional networks for relation classification. BERT+MTB (Baldini Soares et al., 2019) trains relation representation by matching the blanks. We also include the baseline models of BERT-base (Zhang et al., 2019), ERNIE (Zhang et al., 2019), BERT-large (Baldini Soares et al., 2019), KnowBERT (Peters et al., 2019), KEPLER (Wang et al., 2019), RoBERTa-



(a) Open Entity (b) CosmosQA (c) TACRED

Figure 2: Ablation study of the SEPREM model on three different datasets over entity typing, question answering, and relation classification tasks. All the evaluation models are pre-trained on 10 million sentences.

Case	Input Sequence	Model	Prediction	
1	Baldino was born May 13 , 1953 , and grew up in New Jersey ···	RoBERTa	per:stateorprovince_of_birth	
		SEPREM	per:stateorprovinces_of_residence	(√)
2	ALICO , a member company of AIG is looking for one J2EE developer ···	RoBERTa	org:parents	
		SEPREM	org:member_of	(√)
3	And strangely enough , Cain 's short , three-year tenure at the NRA is ···	RoBERTa	no_relation	
		SEPREM	org:top_members/employees	(√)

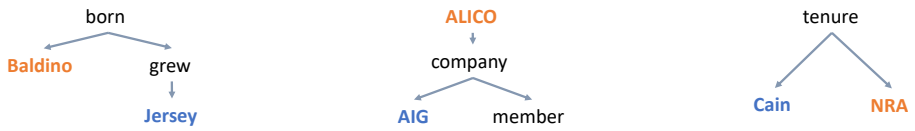


Figure 3: Case study results on the TACRED dataset of relation classification tasks. Models are required to predict the relation between tokens in **orange** and **blue** colors. Predictions with mark ✓are the same with true labels.

large, RoBERTa-large (continue training), and K-Adapter (Wang et al., 2020) for a comprehensive comparison.

Experimental Results Table 5 shows the performances of baseline models and the proposed SEPREM on TACRED. As we can see that the proposed syntax-aware pre-training tasks and syntax-aware attention mechanism can continuously bring gains in relation classification task and SEPREM outperforms baseline models overall. This further confirms the outstanding generalization capacity of our proposed model. It can be also seen that compared with K-Adapter model, the performance gains of SEPREM model observed in the TACRED dataset are not as substantial as that in Open Entity dataset. This may be partially due to the fact that K-Adapter also injects factual knowledge into the model, which may help in identifying relationships.

5.4 Ablation Study

To investigate the impacts of various components in SEPREM, experiments are conducted for entity typing, question answering and relation classification tasks under the different corresponding benchmarks, $i.e.$, Open Entity, CosmosQA, and TACRED, respectively. Note that due to the time-consuming issue of training the models on entire data, we randomly sample 10 million sentences from the whole data to build a small dataset in this ablation study.

The results are illustrated in Figure 2, in which we eliminate two syntax-aware pre-training tasks (*i.e.,* HP and DP) and syntax-aware attention layer to evaluate their effectiveness. It can be seen that without using the syntax-aware attention layer, immediate performance degradation is observed, indicating that leveraging syntax-aware attention layer to learn syntax-aware representation could benefit the SEPREM. Another observation is that for all three experiments, eliminating DP pre-training task leads to worse empirical results. In other words, compared with existing method (*i.e.*, head prediction task), the proposed dependency distance prediction task is more advantageous to various downstream tasks. This observation may be attributed

to the fact that leveraging global syntactic correlation is more beneficial than considering local correlation. Moreover, significant performance gains can be obtained by simultaneously exploiting the two pre-training tasks and syntax-aware attention layer, which further confirms superiority of our pre-training architecture.

5.5 Case Study

We conduct a case study to empirically explore the effectiveness of utilizing syntax information. In the case of relation classification task, we need to predict the relationship of two tokens in a sentence. As the three examples shown in Figure 3, SEPREM can capture the syntax information by the dependency tree and make correct predictions. However, without utilizing syntax information, RoBERTa fails to recognize the correct relationship. To give further insight of how syntax information affects prediction, we also take case 1 for detailed analysis. The extracted dependency tree captures the close correlation of "*grew*" and "*Jersey*", which indicates that "*New Jersey*" is more likely to be a residence place. These results reflects that our model can better understand the global syntax relations among tokens by utilizing dependency tree.

5.6 Analysis of Importance Score α

Under the syntax-enhanced pre-trained framework introduced here, the contextual representation ($\boldsymbol{H}^n$) and syntax-aware representation ($\hat{H}^n$) are jointly optimized to abstract semantic information from sentences. An interesting question concerns how much syntactic information should be leveraged for our pre-trained model. In this regard, we further investigate the effect of the importance score α on the aforementioned six downstream tasks, and the learned weights α after fine-tuning SEPREM model are shown in Table 6. We observe that the values of α are in the range of 13% and 15% on six downstream datasets, which indicates that those downstream tasks require syntactic information to obtain the best performance and once again confirms the effectiveness of utilizing syntax information.

To have a further insight of the effect brought by importance score α, we conduct experiments on SEPREM w/o α, which eliminates the α in Equation 1 and equally integrates the syntax-aware and contextual representation, i.e., $\boldsymbol{H}^n = transformer_n(\boldsymbol{H}^{n-1} + \hat{\boldsymbol{H}}^{n-1})$. The pre-training settings of the SEPREM w/o α model are the same with the proposed SEPREM model. It can be seen in Table 6 that, the performances drop 1%∼3% on the six datasets when excluding the α. This observation indicates the necessity of introducing the α to better integrate the syntax-aware and contextual representation.

Datasets	Model	Performance	Values of α
Open Entity	SEPREM	79.06	0.1334
	SEPREM w/o α	77.13	-
FIGER	SEPREM	82.05	0.1428
	SEPREM w/o α	79.54	-
SearchQA	SEPREM	67.74	0.1385
	SEPREM w/o α	66.31	-
Quasar-T	SEPREM	53.18	0.1407
	SEPREM w/o α	51.84	-
CosmosQA	SEPREM	82.37	0.1357
	SEPREM w/o α	81.06	-
TACRED	SEPREM	72.42	0.1407
	SEPREM w/o α	71.82	-

Table 6: The model's performance and the corresponding values of importance score α after fine-tuning on six public benchmark datasets. Performance is under the evaluate metrics of either Mi-F1 or accuracy scores.

6 Conclusion

In this paper, we present SEPREM that leverage syntax information to enhance pre-trained models. To inject syntactic information, we introduce a syntax-aware attention layer and a newly designed pre-training task are proposed. Experimental results show that our method achieves state-of-the-art performance over six datasets. Further analysis shows that the proposed dependency distance prediction task performs better than dependency head prediction task.

Acknowledgments

We are grateful to Yeyun Gong, Ruize Wang and Junjie Huang for fruitful comments. We are obliged to Zijing Ou and Wenxuan Li for perfecting this article. We appreciate Genifer Zhao for beautifying the figures of this article. Zenan Xu and Qinliang Su are supported by the National Natural Science Foundation of China (No. 61806223, 61906217, U1811264), Key R&D Program of Guangdong Province (No. 2018B010107005), National Natural Science Foundation of Guangdong Province (No. 2021A1515012299). Zenan Xu and Qinliang Su are also supported by Huawei MindSpore.

References

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *ACL*, pages 2895–2905.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Andrea Gesmundo, Neil Houlsby, Wojciech Gajewski, and Wei Wang. 2018. Ask the Right Questions: Active Question Reformulation with Reinforcement Learning. In *ICLR*.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *ACL*, pages 87–96.

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.

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading. In *arXiv preprint arXiv:1707.03904*.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A New Q&A Dataset Augmented with Context from a Search Engine. In *ArXiv preprint arXiv:1704.05179*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *EMNLP*, pages 2391–2401.

Dan Jurafsky. 2000. *Speech & language processing*. Pearson Education India.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *ACL*, pages 1736–1745.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *TACL*, 3:315–328.

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*.

Xuan-Phi Nguyen, Shafiq Joty, Steven Hoi, and Richard Socher. 2020. Tree-structured attention with hierarchical accumulation. In *International Conference on Learning Representations*.

Matthew E Peters, Mark Neumann, IV Logan, L Robert, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *EMNLP*, pages 43–54.

F. Petroni, Tim Rocktäschel, Patrick Lewis, A. Bakhtin, Y. Wu, Alexander H. Miller, and S. Riedel. 2019. Language models as knowledge bases? *ArXiv*, abs/1909.01066.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. Bert is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *ArXiv*, abs/1911.03681.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Devendra Singh Sachan, Yuhao Zhang, Peng Qi, and William Hamilton. 2020. Do syntax trees help pre-trained transformers extract information? *arXiv preprint arXiv:2008.09084*.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017*.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction(AKBC)*, pages 69–74.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. olmpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

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.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Cuihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2017a. Reinforced reader-ranker for open-domain question answering. *arXiv preprint arXiv:1709.00023*.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018. Evidence aggregation for answer re-ranking in open-domain question answering. In *ICLR*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017b. Gated self-matching networks for reading comprehension and question answering. In *ACL*, pages 189–198.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2019. Kepler: A unified model for knowledge embedding and pre-trained language representation. *arXiv preprint arXiv:1911.06136*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. In *ICLR*.

Wang Yaushian, Lee Hung-Yi, and Chen Yun-Nung. 2019. Multitree transformer: Integrating tree structures into self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 9054–9065. Curran Associates, Inc.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In *EMNLP*, 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 *EMNLP*, pages 35–45.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *ACL*, pages 1441–1451.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, and Sufeng Duan. 2020. Sg-net: Syntax-guided machine reading comprehension. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-2020)*.