Retrofitting Structure-aware Transformer Language Model for End Tasks

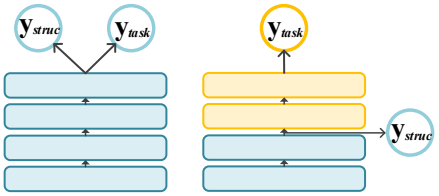
Hao Fei[1], **Yafeng Ren**[2*] and **Donghong Ji**[1]
1. Department of Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, China
2. Guangdong University of Foreign Studies, China
{hao.fei,renyafeng,dhji}@whu.edu.cn

Abstract

We consider retrofitting structure-aware Transformer language model for facilitating end tasks by proposing to exploit syntactic distance to encode both the phrasal constituency and dependency connection into the language model. A middle-layer structural learning strategy is leveraged for structure integration, accomplished with main semantic task training under multi-task learning scheme. Experimental results show that the retrofitted structure-aware Transformer language model achieves improved perplexity, meanwhile inducing accurate syntactic phrases. By performing structure-aware fine-tuning, our model achieves significant improvements for both semantic- and syntactic-dependent tasks.

1 Introduction

Natural language models (LM) can generate fluent text and encode factual knowledge (Mikolov et al., 2013; Pennington et al., 2014; Merity et al., 2017). Recently, pre-trained contextualized language models have given remarkable improvements on various NLP tasks (Peters et al., 2018; Radford et al., 2018; Howard and Ruder, 2018; Yang et al., 2019; Devlin et al., 2019; Dai et al., 2019). Among such methods, the Transformer-based (Vaswani et al., 2017) BERT has become a most popular encoder for obtaining state-of-the-art NLP task performance. It has been shown (Conneau et al., 2018; Tenney et al., 2019) that besides rich semantic information, implicit language structure knowledge can be captured by a deep BERT (Vig and Belinkov, 2019; Jawahar et al., 2019; Goldberg, 2019). However, such structure features learnt via the vanilla Transformer LM are insufficient for those NLP tasks that heavily rely on syntactic or linguistic knowledge (Hao et al., 2019). Some effort devote to improved the ability of structure learning in Transformer LM by installing novel syntax-attention mechanisms (Ahmed et al., 2019; Wang et al., 2019). Nevertheless, several limitations can be observed.



(a) Full-layer (b) Middle-layer

Figure 1: Full-layer multi-task learning for structural training (left), and the middle-layer training for deep structure-aware Transformer LM (right).

First, according to the recent findings by probing tasks (Conneau et al., 2018; Tenney et al., 2019; Goldberg, 2019), the syntactic structure representations are best retained right at the middle layers (Vig and Belinkov, 2019; Jawahar et al., 2019). Nevertheless, existing tree Transformers employ traditional full-scale training over the whole deep Transformer architecture (as shown in Figure 1(a)), consequently weakening the upper-layer semantic learning that can be crucial for end tasks. Second, these tree Transformer methods encode either standalone constituency or dependency structure, while different tasks can depend on varying types of structural knowledge. The constituent and dependency representation for syntactic structure share underlying linguistic characteristics, while the former focuses on disclosing phrasal continuity and the latter aims at indicating dependency relations among elements. For example, semantic parsing tasks are more dependent on the dependency features (Rabinovich et al., 2017; Xia et al., 2019), while constituency information is much needed for sentiment classification (Socher et al., 2013).

In this paper, we aim to retrofit structure-aware

*Corresponding author.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 2151–2161,
November 16–20, 2020.

Transformer LM for facilitating end tasks. • On the one hand, we propose a structure learning module for Transformer LM, meanwhile exploiting syntactic distance as the measurement for encoding both the phrasal constituency and the dependency connection. • On the other hand, as illustrated in Figure 1, to better coordinate the structural learning and semantic learning, we employ a middle-layer structural training strategy to integrate syntactic structures to the main language modeling task under multi-task scheme, which encourages the induction of structural information to take place at most suitable layer. • Last but not least, we consider performing structure-aware fine-tuning with end-task training, allowing learned syntactic knowledge in accordance most with the end task needs.

We conduct experiments on language modeling and a wide range of NLP tasks. Results show that the structure-aware Transformer retrofitted via our proposed middle-layer training strategy achieves better language perplexity, meanwhile inducing high-quality syntactic phrases. Besides, the LM after structure-aware fine-tuning can give significantly improved performance for various end tasks, including semantic-dependent and syntactic-dependent tasks. We also find that supervised structured pre-training brings more benefits to syntactic-dependent tasks, while the unsupervised LM pre-training brings more benefits to semantic-dependent tasks. Further experimental results on unsupervised structure induction demonstrate that different NLP tasks rely on varying types of structure knowledge as well as distinct granularity of phrases, and our retrofitting method can help to induce structure phrases that are most adapted to the needs of end tasks.

2 Related Work

Contextual language modeling. Contextual language models pre-trained on a large-scale corpus have witnessed significant advances (Peters et al., 2018; Radford et al., 2018; Howard and Ruder, 2018; Yang et al., 2019; Devlin et al., 2019; Dai et al., 2019). In contrast to the traditional static and context-independent word embedding, contextual language models can strengthen word representations by dynamically encoding the contextual sentences for each word during pre-training. By further fine-tuning with end tasks, the contextualized word representation from language models can help to give the most task-related context-sensitive features (Peters et al., 2018). In this work, we follow the line of Transformer-based (Vaswani et al., 2017) LM (e.g., BERT), considering its prominence.

Structure induction. The idea of introducing tree structures into deep models for structure-aware language modeling has long been explored by supervised structure learning, which generally relies on annotated parse trees during training and maximizes the joint likelihood of sentence-tree pairs (Socher et al., 2010, 2013; Tai et al., 2015; Yazdani and Henderson, 2015; Dyer et al., 2016; Alvarez-Melis and Jaakkola, 2017; Aharoni and Goldberg, 2017; Eriguchi et al., 2017; Wang et al., 2018; Gū et al., 2018).

There has been much attention paid to unsupervised grammar induction task (Williams et al., 2017; Shen et al., 2018a,b; Kuncoro et al., 2018; Kim et al., 2019a; Luo et al., 2019; Drozdov et al., 2019; Kim et al., 2019b). For example, PRPN (Shen et al., 2018a) computes the syntactic distance of word pairs. On-LSTM (Shen et al., 2018b) allows hidden neurons to learn long-term or short-term information by a gate mechanism. URNNG (Kim et al., 2019b) applies amortized variational inference, encouraging the decoder to generate reasonable tree structures. DIORA (Drozdov et al., 2019) uses inside-outside dynamic programming to compose latent representations from all possible binary trees. PCFG (Kim et al., 2019a) achieves grammar induction by probabilistic context-free grammar. Unlike these recurrent network based structure-aware LM, our work focuses on structure learning for a deep Transformer LM.

Structure-aware Transformer language model. Some efforts have been paid for the Transformer-based pre-trained language models (e.g. BERT) by visualizing the attention (Vig and Belinkov, 2019; Kovaleva et al., 2019; Hao et al., 2019) or probing tasks (Jawahar et al., 2019; Goldberg, 2019). They find that the latent language structure knowledge is best retained at the middle-layer in BERT (Vig and Belinkov, 2019; Jawahar et al., 2019; Goldberg, 2019). Ahmed et al. (2019) employ a decomposable attention mechanism for recursively learn the tree structure for Transformer. Wang et al. (2019) integrate tree structures into Transformer via constituency-attention. However, these Transformer LMs suffer from the full-scale structural training and monotonous types of the structure, limiting the performance of structure LMs for end

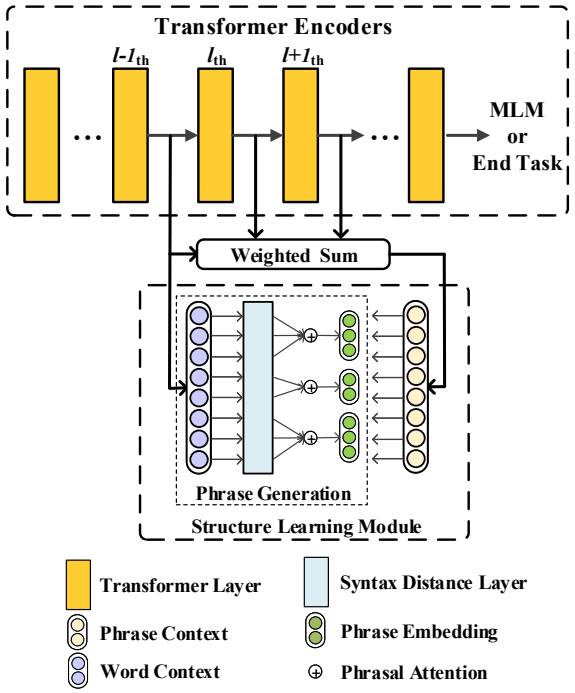


Figure 2: Overall framework of the retrofitted structure-aware Transformer language model.

tasks. Our work is partially inspired by Shen et al. (2018a) and Luo et al. (2019) on employing syntax distance measurements, while their works focus on the syntax learning by recurrent LMs.

3 Model

The proposed structure-aware Transformer language model mainly consists of two components: the Transformer encoders and structure learning module, which are illustrated in Figure 2.

3.1 Transformer Encoder

The language model is built based on N-layer Transformer blocks. One Transformer layer applies multi-head self-attention in combination with a feedforward network, layer normalization and residual connections. Specifically, the attention weights are computed in parallel via:

$$\begin{aligned} \boldsymbol{E} &= \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}})\boldsymbol{V} \\ &= \text{softmax}(\frac{(t\cdot\boldsymbol{x})\quad(t\cdot\boldsymbol{x})}{\sqrt{d}})(t\cdot\boldsymbol{x}) \end{aligned} \tag{1}$$

where Q (query), K (key) and V (value) in multi-head setting process the input $\boldsymbol{x} = \{x_1, \cdots, x_n\}$ t times.

Given an input sentence $\boldsymbol{x}$, the output contextual representation of the l-th layer Transformer block

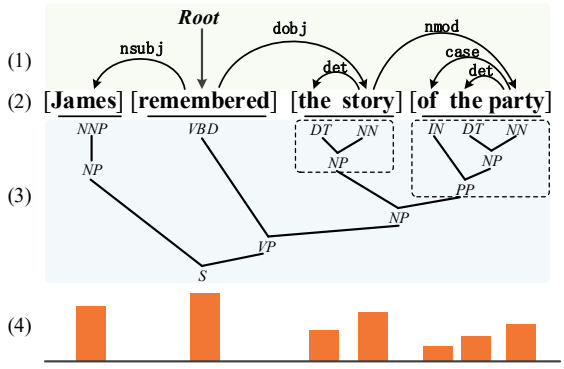


Figure 3: Simultaneously measuring dependency relations (1) and phrasal constituency (3) based on the example sentence (2) by employing syntax distance (4).

can be formulated as:

$$\begin{aligned} \{\boldsymbol{h}_1^l, \cdots, \boldsymbol{h}_n^l\} &= \text{Trm}(\{x_1, \cdots, x_n\}) \\ &= \eta(\Phi(\eta(\boldsymbol{E}^l)) + \boldsymbol{E}^l) \end{aligned} \tag{2}$$

where η is the layer normalization operation and Φ is a feedforward network. In this work, the output contextual representation $\boldsymbol{h}^l = \{\boldsymbol{h}_1^l, \cdots, \boldsymbol{h}_n^l\}$ of the middle layers can be used to learn the structure y_{struc}, and the one at the final layer will be used for language modeling or end task training y_{task}.

3.2 Unsupervised Syntax Learning Module

The structure learning module is responsible for unsupervisedly generating phrases, providing structure-aware language modeling to the host LM.

Syntactic context. We extract the context representations from Transformer middle layers for the next syntax learning. We optimize the structure-aware Transformer LM by forcing the structure knowledge injection focused at middle three layers: $(l-1)_{\text{th}}$, l_{th}, and $(l+1)_{\text{th}}$. Note that although we only make structural attending to the selected layers, structure learning can enhance lower layers via back-propagation.

Specifically, we take the first of the chosen three-layer as the word context $\boldsymbol{C}^\Psi = \boldsymbol{h}^{l-1}$. For the phrasal context $\boldsymbol{C}^\Omega = \{\boldsymbol{c}_1^\Omega, \cdots, \boldsymbol{c}_n^\Omega\}$, we make use of contextual representations from the three chosen layers by weighted sum:

$$\boldsymbol{C}^\Omega = \alpha_{l-1}\cdot\boldsymbol{h}^{l-1} + \alpha_l\cdot\boldsymbol{h}^l + \alpha_{l+1}\cdot\boldsymbol{h}^{l+1} \tag{3}$$

where α_{l-1}, α_l and α_{l+1} are sum-to-one trainable coefficients. Rich syntactic representations are expected to be captured in $\boldsymbol{C}^\Omega$ by LM.

Structure measuring. In this study, we reach the goal of measuring syntax by employing ***syntax distance***. The general concept of syntax distance d_i can be reckoned as a metric (i.e., distance) from a certain word x_i to the root node within the dependency tree (Shen et al., 2018a). For instance in Figure 3, the head word '*remembered*' x_i and its dependent word '*James*' x_j follow $d_i < d_j$. While in this work, to maintain both the dependency and phrasal constituents simultaneously, we add additional constraints on words and phrases. Given two words x_i and x_j $(0 \leq i < j \leq n)$ in one phrase, we define $d_i < d_j$. This can be demonstrated by the word pair '*the*' and '*story*'. While if they are in different phrases[1], e.g., S_u and S_v, the corresponding inner-phrasal head words follow d_i (in S_u) $> d_j$ (in S_V), e.g., '*story*' and '*party*'.

In the structure learning module, we first compute the syntactic distances $\boldsymbol{d} = \{d_1, \cdots, d_n\}$ for each word based on the word context via a convolutional network:

$$\{d_1, \cdots, d_n\} = \Phi(\text{CNN}(\{\boldsymbol{c}_1^{\Psi}, \cdots, \boldsymbol{c}_n^{\Psi}\})) \quad (4)$$

where d_i is a scalar, and Φ is for linearization. With such syntactic distance, we expect both the dependency as well as constituency syntax can be well captured in LM.

Syntactic phrase generating. Considering the word x_i opening an induced phrase $S_m = [x_i, \cdots, x_{i+w}]$ in a sentence, where w is the phrase width, we need to decide the probability $p^*(x_j)$ that a word x_j (j=$i + w + 1$) (i.e., the first word outside phrase S_m) belongs to S_m:

$$p^*(x_j) = \prod_{k=i}^{i+w} \text{sigmoid}(d_j - d_k). \quad (5)$$

We set the initial width $w = 1$, if $p^*(x_j)$ is above the window threshold λ, x_j should be considered inside the phrase; otherwise, the phrase S_m should be closed and restart at x_j. We incrementally conduct such phrasal searching procedure to segment all the phrases in a sentence. Given an induced phrase $S_m = [x_i, \cdots, x_{i+w}]$, we obtain its embedding $\boldsymbol{s}_m$ via a phrasal attention:

$$u_i = \text{softmax}(d_i \cdot p^*(x_i)) \quad (6)$$

$$\boldsymbol{s}_m = \sum_{i}^{i+w} u_i \cdot \boldsymbol{c}_i^{\Psi} \quad (7)$$

[1]Note that we cannot explicitly define the granularity (width) of every phrases in constituency tree, while instead it will be decided by the structure learning module in heuristics.

4 Structure-aware Learning

Multi-task training for language modeling and structure induction. Different from traditional language models, a Transformer-based LM employs the masked language modeling (MLM), which can capture larger contexts. Likewise, we predict a masked word using the corresponding context representation at the top layer:

$$p^{\text{W}}(y_i|\boldsymbol{x}) = \text{softmax}(\boldsymbol{c_i}|\boldsymbol{x}) \quad (8)$$

$$\mathcal{L}_{\text{W}} = \sum_{i}^{k} \log p^{\text{W}}(y_i|\boldsymbol{x}) \quad (9)$$

On the other hand, the purpose on unsupervised syntactic induction is to encourage the model to induce $\boldsymbol{s}_m$ that is most likely entailed by the phrasal context $\boldsymbol{c}_i^{\Omega}$. The behind logic lies is that, if the initial Transformer LM can capture linguistic syntax knowledge, then after iterations of learning with the structure learning module, the induced structure can be greatly amplified and enhanced (Luo et al., 2019). We thus define the following probability:

$$p^{\text{G}}(\boldsymbol{s}_m|\boldsymbol{c}_i^{\Omega}) = \frac{1}{1 + \exp(-\boldsymbol{s}_m^T \cdot \boldsymbol{c}_i^{\Omega})} \quad (10)$$

Additionally, to enhance the syntax learning, we employ negative sampling:

$$\mathcal{L}_{\text{Neg}} = \frac{1}{n} \sum_{j}^{n} p^{\text{G}}(\hat{\boldsymbol{s}}_j^T|\boldsymbol{c}_i^{\Omega}) \quad (11)$$

where $\hat{s}$ is a randomly selected negative phrase. The final objective for structure learning is:

$$\mathcal{L}_{\text{G}} = \sum_{i}^{K}(\sum_{m}^{\text{M}}(1 - p^{\text{G}}(\boldsymbol{s}_m|\boldsymbol{c}_i^{\Omega})) + \mathcal{L}_{\text{Neg}}) \quad (12)$$

We employ multi-task learning for simultaneously training our LM for both word prediction and structure induction. Thus, the overall target is to minimize the following multi-task loss objective:

$$\mathcal{L}_{\text{pre}} = \mathcal{L}_{\text{W}} + \gamma^{\text{pre}} \cdot \mathcal{L}_{\text{G}} \quad (13)$$

where γ^{pre} is a regulating coefficient.

Supervised syntax injection. Our default structure-aware LM unsupervisedly induces syntax at the pre-training stage, as elaborated above. Alternatively, in Eq. (7), if we leverage the gold (or apriori) syntax distance information for phrases, we can achieve supervised structure injection.

Unsupervised structure fine-tuning. We aim to improve the learnt structural information for better facilitating the end tasks. Therefore, during the fine-tuning stage of end tasks, we consider further making the structure learning module trainable:

$$\mathcal{L}_{\text{fine}} = \mathcal{L}_{\text{task}} + \gamma^{\text{fine}} \cdot \mathcal{L}_{\text{G}} \quad (14)$$

where $\mathcal{L}_{\text{task}}$ refers to the loss function of the end task, and γ^{fine} is a regulating coefficient. Note that to achieve the best structural fine-tuning, the supervised structure injection is unnecessary, and we do not allow supervised structure aggregation at the fine-tuning stage.

Our approach is model-agnostic as we realize the syntax induction via a standalone structure learning module, which is disentangled from a host LM. Thus the method can be applied to various Transformer-based LM architectures.

5 Experiments

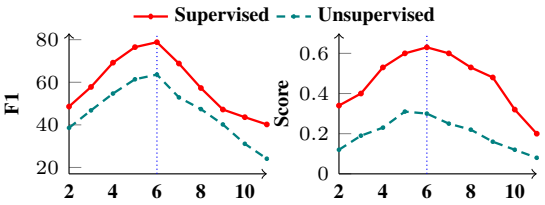
5.1 Experimental Setups

We employ the same architecture as BERT base model[2], which is a 12-layer Transformer with 12 attention heads and 768 dimensional hidden size. To enrich our experiments, we also consider the Google pre-trained weights as the initialization. We use Adam as our optimizer with an initial learning rate in [8e-6, 1e-5, 2e-5, 3e-5], and a L2 weight decay of 0.01. The batch size is selected in [16,24,32]. We set the initial values of coefficients α_{l-1}, α_l and α_{l+1} as 0.35, 0.4 and 0.25, respectively. The pre-training coefficient γ^{pre} is set as 0.5, and the fine-tuning one γ^{fine} as 0.23. These values give the best effects in our development experiments. Our implementation is based on the PyTroch library[3].

Besides, for supervised structure learning in our experiments, we use the state-of-the-art BiAffine dependency parser (Dozat and Manning, 2017) to parse sentences for all the relevant datasets, and use the Self-Attentive parser (Kitaev and Klein, 2018) to obtain the constituency structure. Being trained on the English Penn Treebank (PTB) corpus (Marcus et al., 1993), the dependency parser has 95.2% UAS and 93.4% LAS, and the constituency parser has 92.6% F1 score. With the auto-parsed annotations, we can calculate the syntax distances (substitute the ones in Eq. 4) and obtain the corresponding phrasal embeddings (in Eq. 7).

[2] https://github.com/google-research/bert

[3] https://pytorch.org/



(a) Constituency phrase parsing. (b) Dependency alignment.

Figure 4: Development experiments on syntactic probing tasks at varying Transformer layer.

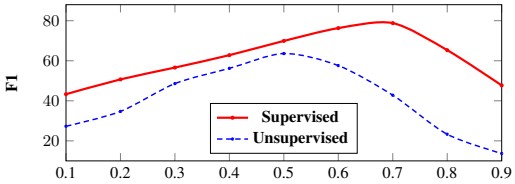


Figure 5: Constituency parsing under different λ.

5.2 Development Experiments

Structural learning layers. We first validate at which layer of depths the structural-aware Transformer LM can achieve the best performance when integrating our retrofitting method. We thus design probing experiments, in which we consider following two syntactic tasks. 1) **Constituency phrase parsing** seeks to generate grammar phrases based on the PTB dataset and evaluate whether induced constituent spans also exist in the gold Treebank dataset. 2) **Dependency alignment** aims to compute the proportion of Transformer attention connecting tokens in a dependency relation (Vig and Belinkov, 2019):

$$\text{Score} = \frac{\sum_{x \in X} \sum_{i=1}^{x} \sum_{j=1}^{x} \alpha_{i,j}(x) \cdot \text{dep}(x_i, x_j)}{\sum_{x \in X} \sum_{i=1}^{x} \sum_{j=1}^{x} \alpha_{i,j}(x)} \quad (15)$$

where $\alpha_{i,j}(x)$ is the attention weight, and $\text{dep}(x_i, x_j)$ is an indicator function (1 if x_i and x_j are in a dependency relation and 0 otherwise). The experiments are based on English Wikipedia, following Vig and Belinkov (2019).

As shown in Figure 4, both the results on unsupervised and supervised phrase parsing are the best at layer 6. Also the attention aligns with dependency relations most strongly in the middle layers (5-6), consistent with findings from previous work (Tenney et al., 2019; Vig and Belinkov, 2019). Both two probing tasks indicate that our proposed middle-layer structure training is practical. We thus inject the structure in the structure learning module at the 6-th layer ($l = 6$).

system	Syntactic.			Semantic.					*Avg.*
	TreeDepth	TopConst	Tense	SOMO	NER	SST	Rel	SRL	
• **w/o Initial Weight:**									
Trm	25.31	40.32	61.06	50.11	89.22	86.21	84.70	88.30	65.65
RvTrm	29.52	45.01	63.83	51.42	89.98	86.66	85.02	88.94	67.55
Tree+Trm	30.37	46.58	65.83	53.08	90.62	87.25	84.97	88.70	68.43
PI+TrmXL	31.28	47.06	63.78	52.36	90.34	87.09	85.22	89.02	68.27
Ours+Trm									
+usp.	33.98	49.69	66.39	57.04	92.24	90.48	87.05	90.87	70.74
+sp.	37.35	57.68	72.04	56.41	91.86	90.06	86.34	90.54	73.12
+syn-embed.	36.28	54.30	67.61	55.68	91.87	87.10	86.87	89.41	71.14
• **Initial Weight:**									
BERT	38.61	79.37	90.61	65.31	92.40	93.50	89.25	92.20	80.16
Ours+BERT(**usp.**)	45.82	88.64	94.68	67.84	94.28	94.67	90.41	93.12	83.68

Table 1: Structure-aware Transformer LM for end tasks.

System	Const.	Ppl.
PRPN	42.8	-
On-LSTM	49.4	-
URNNG	52.4	-
DIORA	56.2	-
PCFG	60.1	-
Trm	22.7	78.6
RvTrm	47.0	50.3
Tree+Trm	52.0	45.7
PI+TrmXL	56.2	43.4
Ours+Trm		
+usp.	60.3	37.0
+sp.	68.8	29.2
BERT	31.3	21.5
Ours+BERT(**usp.**)	65.2	16.2

Table 2: Performance on constituency parsing and language modeling.

Phrase generation threshold. We introduce a hyper-parameter λ as a threshold to decide whether a word belong to a given phrase during the phrasal generation step. We explore the best λ value based on the same parsing tasks. As shown in Figure 5, with $\lambda = 0.5$ for unsupervised induction and $\lambda = 0.7$ for supervised induction, the induced phrasal quality is the highest. Therefore we set such λ values for all the remaining experiments.

5.3 Structure-aware Language Modeling

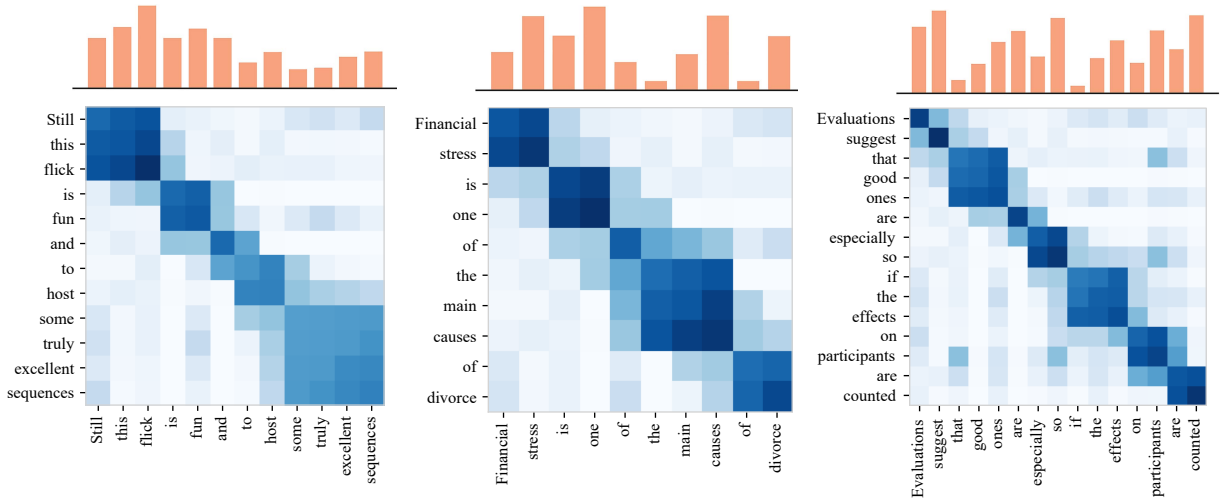
We evaluate the effectiveness of our proposed retrofitted structure-aware LM after pre-training. We first compare the performance on language modeling[4]. From the results shown in Table 2, our retrofitted Transformer yields better language perplexity in both unsupervised (37.0) or supervised (29.2) manner. This proves that our middle-layer structure training strategy can effectively relieve negative mutual influence of structure learning on semantic learning, while inducing high-quality of structural phrases. We can also conclude that language models with more successful structural knowledge can better help to encode effective intrinsic language patterns, which is consistent with the prior studies (Kim et al., 2019b; Wang et al., 2019; Drozdov et al., 2019).

We also compare the constituency parsing with state-of-the-art structure-aware models, including **1) Recurrent-based models** described in §2: PRPN (Shen et al., 2018a), On-LSTM (Shen et al., 2018b), URNNG (Kim et al., 2019b), DIORA (Drozdov et al., 2019), PCFG (Kim et al., 2019a), and **2) Transformer based methods**: Tree+Trm (Wang et al., 2019), RvTrm (Ahmed et al., 2019), PI+TrmXL (Luo et al., 2019), and the BERT model initialized with rich weights. As shown in Table 2, all the structure-aware models can give good parsing results, compared with non-structured models. Our retrofitted Transformer LM gives the best performance (60.3% F1) in unsupervised induction. Combined with the supervised auto-labeled parses, it give the highest F1 score (68.8%).

5.4 Fine-tuning for End Tasks

We validate the effectiveness of our method for end tasks with structure-aware fine-tuning. All systems are first pre-trained for structure learning, and then fine-tuned with end task training. The evaluation is performed on eight tasks, involving

[4]Transformer can see its subsequent words bidirectionally, so we measure the perplexity on masked words. And we thus avoid directly comparing with the Recurrent-based LMs.



(a) SST (b) Rel (c) SRL

Figure 6: Visualization of attention heads (heatmap) and the corresponding syntax distances (bar chart).

syntactic tasks and semantic tasks. `TreeDepth` predicts the depth of the syntactic tree, `TopConst` tests the sequence of top level constituents in the syntax tree, and `Tense` detects the tense of the main-clause verb, while `SOMO` checks the sensitivity to random replacement of words, which are the standard probing tasks. We follow the same datasets and settings with previous work (Conneau et al., 2018; Jawahar et al., 2019).

Also we evaluate the semantic tasks including 1) `NER`, named entity recognition on CoNLL03 (Tjong Kim Sang and De Meulder, 2003), 2) `SST`, binary sentiment classification task on Standford sentiment treebank (Socher et al., 2013), 3) `Rel`, relation classification on Semeval10 (Hendrickx et al., 2010), and 4) `SRL`, semantic role labeling task on the CoNLL09 WSJ (Hajič et al., 2009). The performance is reported by the F1 score.

The results are summarized in Table 1. First, we find that structure-aware LMs bring improved performance for all the tasks, compared with the vanilla Transformer encoder. Second, the Transformer with our structural-aware fine-tuning achieves better results (70.74% on average) for all the end tasks, compared with the baseline tree Transformer LMs. This proves that our proposed middle-layer strategy best benefits the structural fine-tuning, compared with the full-layer structure training on baselines. Third, with supervised structure learning, significant improvements can be found across all tasks.

For the supervised setting, we replace the supervised syntax fusion in structure learning module with the auto-labeled syntactic dependency embedding and concatenate it with other input embeddings. The results are not as prominent as the supervised syntax fusion, which reflects the advantage of our proposed structure learning module. Besides, based on the task improvements from the retrofitted Transformer by our method, we can further infer that the supervised structure benefits more syntactic-dependent tasks, and the unsupervised structure benefits semantic-dependent tasks the most. Finally, the BERT model integrating with our method can give improved effects[5].

	Mean	Median
RvTrm	0.68	0.69
Tree+Trm	0.60	0.64
PI+TrmXL	0.54	0.58
Ours+Trm(**usp.**)	0.50	0.52
Ours+Trm(**sp.**)	0.32	0.37

Table 3: Fine-grained parsing.

6 Analysis

6.1 Induced Phrase after Pre-training.

We take a further step, evaluating the fine-grained quality on phrasal structure induction after pre-training. Instead of checking whether the induced constituent spans are identical to the gold counterparts, we now consider measuring the deviation $PhrDev(\hat{y}, y) = \sqrt{\frac{1}{N}\sum_i [\Delta(\hat{y}_i, y_i) - \overline{\Delta}]^2}$,

[5]We note that the direct comparison with BERT model is not fair, because the large numbers of well pre-trained parameters can bring overwhelming advances.

where $\Delta(\hat{y}_i, y_i)$ is the phrasal editing distance between the induced phrase length and the gold length within a sentence. $\overline{\Delta}$ is the averaged editing distance. If all the predicted phrases are same with the ground truth, or all different from it, $PhrDev(\hat{y}, y) = 0$, which means that the phrases are induced with the maximum consistency, and vice versa. We make statistics for all the sentences in Table 3. Our method can unsupervisedly generate higher quality of structural phrases, while we can achieve the best injection of the constituency knowledge into LM by the supervised manner.

6.2 Fine-tuned Structures with End Tasks

Interpreting fine-tuned syntax. To interpret the fine-tuned structures, we empirically visualize the Transformer attention head from the chosen l-layer, and the syntax distances of the sentence. We exhibit three examples from SST, Rel and SRL, respectively, as shown in Figure 6. Overall, our method can help to induce clear structure of both dependency and constituency. While interestingly, different types of tasks rely on different granularity of phrase. Comparing the heat maps and syntax distances with each other, the induced phrasal constituency on SST are longer than that on SRL. This is because the sentiment classification task demands more phrasal composition features, while the SRL task requires more fine-grained phrases. In addition, we find that the syntax distances in SRL and Rel are higher in variance, compared with the ones on SST, Intuitively, the larger deviation of syntax distances in a sentence indicates the more demand to the interdependent information between elements, while the smaller deviation refers to phrasal constituency. This reveals that SRL and Rel rely more on the dependency syntax, while SST is more relevant to constituents, which is consistent with previous studies (Socher et al., 2013; Rabinovich et al., 2017; Xia et al., 2019; Fei et al., 2020).

Distributions of heterogeneous syntax for different tasks. Based on the above analysis, we further analyze the distributions of dependency and constituency structures after fine-tuning, in different tasks. Technically, we calculate the mean absolute differences of syntax distances between elements x_i and the sub-root node x_r in a sentence: $Diff = \frac{1}{N}\sum_i^N |d_i - d_r|$. We then linearly normalize them into [0,1] for all the sentences in the corpus of each task, and make statistics, as plotted in Figure 7. Intuitively, the larger the value is, the more interdependent to dependency syntax the task is, and otherwise, to constituency structure. Overall, distributions of dependency structures and phrasal constituents in fine-tuned LM vary among different tasks, verifying that different tasks depend on distinct types of structural knowledge. For example, TreeDepth, Rel and SRL are most supported by dependency structure, while TopConst and SST benefit from constituency the most. SOMO and NER can gain from both two types.

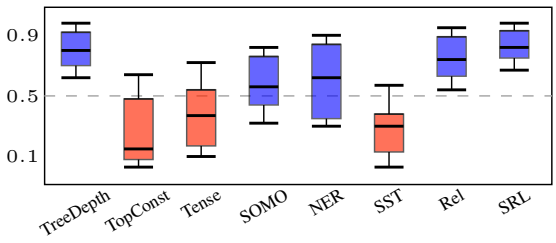


Figure 7: Distributions of dependency and constituency syntax in different tasks. Blue color indicates the predominance of dependency, while Red for constituency.

	SST		SRL	
	Ours+Trm	Tree+Trm	Ours+Trm	Tree+Trm
NP	0.48	0.45	0.37	0.53
VP	0.21	0.28	0.36	0.21
PP	0.08	0.14	0.17	0.06
ADJP	0.10	0.05	0.05	0.12
ADVP	0.07	0.02	0.03	0.02
Other	0.06	0.06	0.02	0.06
Avg.Len.	3.88	3.22	2.69	3.36

Table 4: Proportion of each type of induced phrase.

Phrase types. Finally, we explore the diversity of phrasal syntax required by two representative end tasks, SST and SRL. We first look into the statistical proportion for different types of induced phrases[6]. As shown in Table 4, our method tends to induce more task-relevant phrases, where the lengths of induced phrases are more variable to the task. Concretely, the fine-tuned structure-aware Transformer helps to generate more *NP* also with longer phrases for the SST task, and yield roughly equal numbers of *NP* and *VP* for SRL tasks with shorter phrases. This evidently gives rise to the better task performance. In contrast, the syntax phrases induced by the Tree+Trm model keep unvarying for SST (3.22) and SRL (3.36) tasks.

[6]Five main types are considered: noun phrase (*NP*), verb phrase (*VP*), prepositional phrase (*PP*), adjective phrase (*ADJP*) and adverb phrase (*ADVP*).

7 Conclusion

We presented a retrofitting method for structure-aware Transformer-based language model. We adopted the syntax distance to encode both the constituency and dependency structure. To relieve the conflict of structure learning and semantic learning in Transformer LM, we proposed a middle-layer structure learning strategy under a multi-tasks scheme. Results showed that structure-aware Transformer retrofitted via our proposed method achieved better language perplexity, inducing high-quality syntactic phrase. Furthermore, our LM after structure-aware fine-tuning gave significantly improved performance for both semantic-dependent and syntactic-dependent tasks, also yielding most task-related and interpretable syntactic structures.

Acknowledgments

We thank the anonymous reviewers for their valuable and detailed comments. This work is supported by the National Natural Science Foundation of China (No. 61772378, No. 61702121), the National Key Research and Development Program of China (No. 2017YFC1200500), the Research Foundation of Ministry of Education of China (No. 18JZD015), the Major Projects of the National Social Science Foundation of China (No. 11&ZD189), the Key Project of State Language Commission of China (No. ZDI135-112) and Guangdong Basic and Applied Basic Research Foundation of China (No. 2020A151501705).

References

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. *CoRR*, abs/1704.04743.

Mahtab Ahmed, Muhammad Rifayat Samee, and Robert E. Mercer. 2019. You only need attention to traverse trees. In *Proceedings of the ACL*, pages 316–322.

David Alvarez-Melis and Tommi S. Jaakkola. 2017. Tree-structured decoding with doubly-recurrent neural networks. In *Proceedings of the ICLR*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the ACL*, pages 2126–2136.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the ACL*, pages 2978–2988.

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 NAACL*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the ICLR*.

Andrew Drozdov, Patrick Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. 2019. Unsupervised latent tree induction with deep inside-outside recursive autoencoders. *CoRR*, abs/1904.02142.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the NAACL*, pages 199–209.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the ACL*, pages 72–78.

Hao Fei, Meishan Zhang, Fei Li, and Donghong Ji. 2020. Cross-lingual semantic role labeling with model transfer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2427–2437.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *CoRR*, abs/1901.05287.

Jetic Gū, Hassan S. Shavarani, and Anoop Sarkar. 2018. Top-down tree structured decoding with syntactic connections for neural machine translation and parsing. In *Proceedings of the EMNLP*, pages 401–413.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the CoNLL*, pages 1–18.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and understanding the effectiveness of BERT. In *Proceedings of the EMNLP*, pages 4141–4150.

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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the ACL*, pages 3651–3657.

Yoon Kim, Chris Dyer, and Alexander Rush. 2019a. Compound probabilistic context-free grammars for grammar induction. In *Proceedings of the ACL*, pages 2369–2385.

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019b. Unsupervised recurrent neural network grammars. In *Proceedings of the NAACL*, pages 1105–1117.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the ACL*, pages 2676–2686.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. *CoRR*, abs/1908.08593.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the ACL*, pages 1426–1436.

Hongyin Luo, Lan Jiang, Yonatan Belinkov, and James Glass. 2019. Improving neural language models by segmenting, attending, and predicting the future. In *Proceedings of the ACL*, pages 1483–1493.

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.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *Proceedings of the ICLR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the NIPS*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the EMNLP*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *in Proceedings of the NAACL*, pages 2227–2237.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the ACL*, pages 1139–1149.

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

Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron C. Courville. 2018a. Neural language modeling by jointly learning syntax and lexicon. In *Proceedings of the ICLR*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. 2018b. Ordered neurons: Integrating tree structures into recurrent neural networks. *CoRR*, abs/1810.09536.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *In Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the EMNLP*, pages 1631–1642.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the ACL*, pages 1556–1566.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *Proceedings of the ICLR*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the CoNLL*, pages 142–147.

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.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. *CoRR*, abs/1906.04284.

Xinyi Wang, Hieu Pham, Pengcheng Yin, and Graham Neubig. 2018. A tree-based decoder for neural machine translation. In *Proceedings of the EMNLP*, pages 4772–4777.

Yaushian Wang, Hung-Yi Lee, and Yun-Nung Chen. 2019. Tree transformer: Integrating tree structures into self-attention. In *Proceedings of the EMNLP*, pages 1061–1070.

Adina Williams, Andrew Drozdov, and Samuel R. Bowman. 2017. Learning to parse from a semantic objective: It works. is it syntax? *CoRR*, abs/1709.01121.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. In *Proceedings of the AAAI*, pages 7305–7313.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Majid Yazdani and James Henderson. 2015. Incremental recurrent neural network dependency parser with search-based discriminative training. In *Proceedings of the CoNLL*, pages 142–152.