Improving Text Understanding via Deep Syntax-Semantics Communication

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`

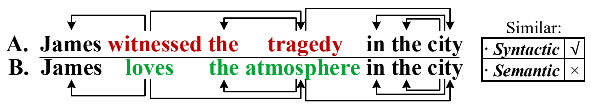
Abstract

Recent studies show that integrating syntactic tree models with sequential semantic models can bring improved task performance, while these methods mostly employ shallow integration of syntax and semantics. In this paper, we propose a deep neural communication model between syntax and semantics to improve the performance of text understanding. Local communication is performed between syntactic tree encoder and sequential semantic encoder for mutual learning of information exchange. Global communication can further ensure comprehensive information propagation. Results on multiple syntax-dependent tasks show that our model outperforms strong baselines by a large margin. In-depth analysis indicates that our method is highly effective in composing sentence semantics.

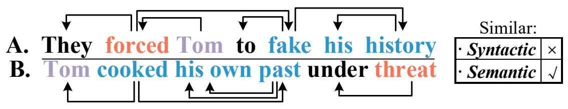
1 Introduction

Neural sequential models such as LSTM (Hochreiter and Schmidhuber, 1997), GRU (Cho et al., 2014), Transformer (Vaswani et al., 2017), ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), have been extensively applied for encoding the semantics of texts in natural language processing (NLP) (Sundermeyer et al., 2012; Bahdanau et al., 2015; Dozat and Manning, 2017; Yuan et al., 2019). On the other hand, hierarchical tree models, such as TreeLSTM (Socher et al., 2013) and GCN (Kipf and Welling, 2017), have been introduced to enrich sequence encoding with syntactic information, bringing further strengths in text modeling. Such external syntactic structure knowledge provides enhanced features, which can facilitate a broad range of NLP tasks (Tai et al., 2015; Looks et al., 2017; Zhang and Zhang, 2019).

Recent studies show that integrating syntactic tree models with sequential semantic models can


A. James witnessed the tragedy in the city
B. James loves the atmosphere in the city
Similar:
· Syntactic √
· Semantic ×

(a) The same syntactic structure but different semantics.


A. They forced Tom to fake his history
B. Tom cooked his own past under threat
Similar:
· Syntactic ×
· Semantic √

(b) The similar semantics but different syntactic structures.

Figure 1: Comparisons of syntax and semantics in sentences. The same color indicates the same (similar) semantic objective.

bring improved performance for syntax-dependent tasks (Shi et al., 2016; Havrylov et al., 2019), such as semantic role labeling (SRL) (Wang et al., 2019) and natural language inference (NLI) (Chen et al., 2017; Liu et al., 2018, 2019), etc. Intuitively, sequential semantic models and syntactic tree models play different roles in text modeling. Sequential semantic models learn the representation via adjacency neighborhood, while syntactic tree models encode texts through structural connections. Taking the two sentence pairs from the NLI task in Figure 1 as example, sentence *A* and *B* in example (a) share the same dependency structure but have irrelevant semantics, and tree models are more suitable and effective for capturing the semantic difference than sequential models in this case. In example (b), two sentences convey very similar semantics but have different syntactic structures. Therefore, two types of models should interact closely in learning compositional representations for better understanding of the texts.

However, existing efforts integrate tree and sequential models through a straightforward way such as representations concatenation (Chen et al., 2017; Vashishth et al., 2019) or multi-task learning (Shi et al., 2016; Swayamdipta et al., 2018; Chen

[*]Corresponding author.

et al., 2019), limiting the performance of end tasks. We believe that a better integration can be achieved when adequate interactions between sequential semantic encoder and syntactic tree encoder can take place during learning, improving the performance of end tasks, and also alleviating the long-range dependency problem.

To this end, we present a novel deep syntax-semantics communication model, as shown in Figure 2. In particular, sequential and dependency-tree based submodels are used for encoding input sentence separately. Local communication is performed between each submodel during learning for information exchange of consecutive words in a sentence. Meanwhile, two submodels are considered as an entire unit, taking global propagation at sentence level over recurrent steps. In addition, we employ gate mechanism to control information flow of each node at each time step during global communication.

Experimental results on a wide range of syntax-dependent NLP tasks show that our model outperforms strong baselines by a large margin, offering an alternative for better integration of sequential and tree models. Further analysis indicates that our method is highly effective in composing sentence semantics, verifying the importance of integrating syntax and semantics for text understanding.

2 Related Work

Neural sequential models have been widely used for encoding texts in the NLP community, due to their effectiveness on capturing semantics. Representative models such as LSTM, GRU, Transformer, ELMo and BERT, have been employed for various NLP tasks, including language modeling (Sundermeyer et al., 2012), machine translation (Bahdanau et al., 2015), question answering (Yuan et al., 2019), etc. On the other hand, some efforts devote to develop hierarchical tree models such as TreeLSTM and GCN, based on syntactic structures (e.g., dependency tree). Such tree encoders equipped with external syntactic knowledge can bring further improvements for some NLP tasks, especially syntax-dependent ones (Tai et al., 2015; Looks et al., 2017; Zhang and Zhang, 2019; Fei et al., 2020b,a), such as SRL (Swayamdipta et al., 2016; Wang et al., 2019; Fei et al., 2020c), NLI (Chen et al., 2017; Liu et al., 2019) and relation classification (Liu et al., 2015; Tran et al., 2019), etc.

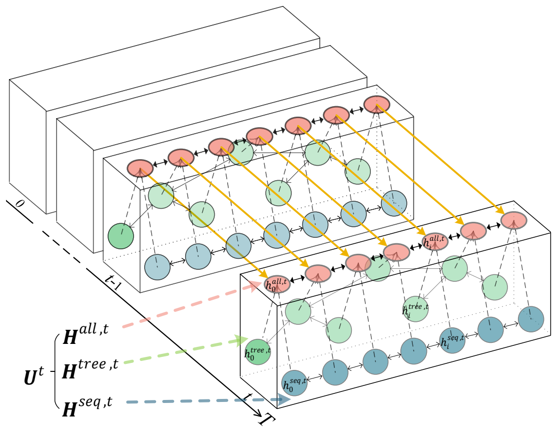


Figure 2: Overall architecture of the proposed model.

In recent years, exploring the correlation between syntax and semantics has become a hot research topic. Previous work has shown a strong correlation between syntax and semantics, and proven that integrating syntactic tree models with sequential models could improve the performance of end tasks (Swayamdipta et al., 2016; Shi et al., 2016; Looks et al., 2017; Liu et al., 2018; Chen et al., 2019). For example, Shi et al. (2016) simultaneously conducted syntax parsing and semantic role labeling via multi-task training strategy. Swayamdipta et al. (2018) incorporated syntactic features into semantic parsing tasks by multi-task learning. Vashishth et al. (2019) concatenated the contextualized semantic representations with syntactic tree representations for improving the ability of word embeddings. More recently, Liu et al. (2019) added a multi-layer BiLSTM with shortcut connections to the *Pairwise Word Interaction* model for capturing semantics and syntactic structure of sentences. However, these methods only use shallow integration of syntax and semantics, limiting the performance of end tasks.

Our model is inspired by Zhang et al. (2019), who introduce a novel method allowing the sufficient communication between different tree models for sentiment analysis. Unlike their work, this paper is dedicated to realizing a deep communication between syntactic tree model and sequential semantic model for improving text understanding. The idea of sentence-level propagation in our work is partially related to Zhang et al. (2018), who propose a novel LSTM architecture where a set of global states are used for sentence-level propagation along recurrent steps, rather than incremental reading of a sequence of words in vanilla sequential LSTM. Compared with their model, our model is more effective in composing semantic information of texts.

3 Model

In this paper, we propose a deep neural communication model between syntax and semantics to improve the performance of text understanding. Local communication is performed between two encoders for information exchange of each node, as illustrated in Figure 3. Global communication is performed over the entire framework under recurrent steps for sufficient information propagation, as shown in Figure 2.

3.1 Baseline Encoders

For an input sentence $S = \{w_1, \cdots, w_n\}$ with sequential word representations $\{x_1^{seq}, \cdots, x_n^{seq}\}$ and dependency tree representations $\{x_1^{tree}, \cdots, x_n^{tree}\}$ from dependency parsing, the baseline sequential encoder and the tree encoder generate contextualized representations separately, which can be concatenated as the final node representation.

3.1.1 Sequential Encoder

We use a bidirectional LSTM (BiLSTM) as sequential encoder on learning semantic information, which processes a sentence in forward and backward directions, based on vanilla LSTM model. Considering the forward node representation $\overrightarrow{h_i}$ from a forward LSTM:

$$i_i = \sigma(\boldsymbol{W}^{(i)} x_i^{\text{seq}} + \boldsymbol{U}^{(i)} + b^{(i)}) \tag{1}$$
$$f_i = \sigma(\boldsymbol{W}^{(f)} x_i^{\text{seq}} + \boldsymbol{U}^{(f)} \overrightarrow{h}_{i-1} + b^{(f)}) \tag{2}$$
$$o_i = \sigma(\boldsymbol{W}^{(o)} x_i^{\text{seq}} + \boldsymbol{U}^{(o)} \overrightarrow{h}_{i-1} + b^{(o)}) \tag{3}$$
$$u_i = \tanh(\boldsymbol{W}^{(u)} x_i^{\text{seq}} + \boldsymbol{U}^{(u)} \overrightarrow{h}_{i-1} + b^{(u)}) \tag{4}$$
$$c_i = i_i \odot u_i + f_i \odot c_{i-1} \tag{5}$$
$$\overrightarrow{h_i} = o_i \odot \tanh(c_i) \tag{6}$$

where i_i, f_i, o_i and u_i are the gates controlling the LSTM cell c_i and the state $\overrightarrow{h_i}$. $\boldsymbol{W}$ and b are the parameters. σ is the sigmoid function and $\odot$ is the element-wise multiplication. Similarly, a backward LSTM can yield the backward node representation $\overleftarrow{h_i}$ over the same input S. BiLSTM takes the concatenation of $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ as the final node representation for the word w_i:

$$h_i^{seq} = [\overrightarrow{h_i}; \overleftarrow{h_i}] \tag{7}$$

3.1.2 Tree Encoder

We employ the dependency tree as the underlying structure, where all the nodes are input words and connected with directed edges, as the sentences shown in Figure 1. We use two typical tree models for encoding the structure, including TreeLSTM and GCN, both under a bidirectional setting.

The standard TreeLSTM encodes each node j with its corresponding head word representation as input x_j. For the bottom-up TreeLSTM:

$$\overline{h}_j^{\uparrow} = \sum_{k \in C(j)} h_k^{\uparrow} \tag{8}$$
$$i_j = \sigma(\boldsymbol{W}^{(i)} x_j^{\text{tree}} + \boldsymbol{U}^{(i)} \overline{h}_j^{\uparrow} + b^{(i)}) \tag{9}$$
$$f_{jk} = \sigma(\boldsymbol{W}^{(f)} x_j^{\text{tree}} + \boldsymbol{U}^{(f)} \overline{h}_k^{\uparrow} + b^{(f)}) \tag{10}$$
$$o_j = \sigma(\boldsymbol{W}^{(o)} x_j^{\text{tree}} + \boldsymbol{U}^{(o)} \overline{h}_j^{\uparrow} + b^{(o)}) \tag{11}$$
$$u_j = \tanh(\boldsymbol{W}^{(u)} x_j^{\text{tree}} + \boldsymbol{U}^{(u)} \overline{h}_j^{\uparrow} + b^{(u)}) \tag{12}$$
$$c_j = i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k \tag{13}$$
$$h_j^{\uparrow} = o_j \odot \tanh(c_j) \tag{14}$$

where $\boldsymbol{W}, \boldsymbol{U}$ and b are parameters, $C(j)$ is the set of child nodes of j. h_j, i_j, o_j and c_j denote the hidden state, input gate, output gate and memory cell of the node j, respectively. f_{jk} is a forget gate for each child k of j. Similarly, the top-down TreeLSTM has the same transitions as the bottom-up counterpart, except for the direction and number of dependent nodes. We use the concatenated representations from two direction for each node:

$$h_j^{tree} = [h_j^{\uparrow}; h_j^{\downarrow}] \tag{15}$$

Compared with TreeLSTM, GCN is more computationally efficient, performing tree propagation for each node in parallel with O(1) complexity. Considering the constructed dependency graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ are sets of nodes and $\mathcal{E}$ are sets of bidirectional edges between heads and dependents, respectively. GCN can be viewed as a hierarchical node encoder, which represents the node j at the k-th layer and encodes the node j as follows:

$$g_j^k = \sigma(\boldsymbol{W}_j^k h_j^k + b_j^k) \tag{16}$$
$$h_j^k = \text{ReLU}(\sum_{j \in \mathcal{N}(j)} x_j^{tree,k} \odot g_j^k) \tag{17}$$

where $\mathcal{N}(j)$ denotes neighbors of j. ReLU is a non-linear activation function. We take the final layer's output as the final tree representation h_j^{tree}.

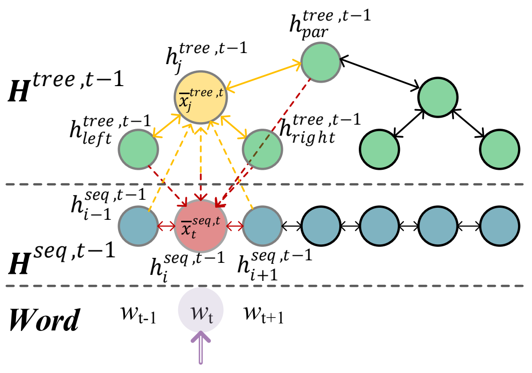


Figure 3: Local communication between sequential encoder and tree encoder.

3.2 Deep Communication Model

We treat sequential encoder and tree encoder as one entire unit, making use of the representation concatenation of the corresponding nodes as the final node representation h_i^{all}:

$$h_i^{all} = [h_i^{seq}; h_i^{tree}]. \tag{18}$$

As shown in Figure 2, during the inner-sentence local interaction between semantics and syntax, we meantime scale the whole unit over recurrent steps T for sentence-level global propagation. We denote the unit at recurrent step t as $\boldsymbol{U}_t = \{\boldsymbol{H}^{seq,t}, \boldsymbol{H}^{tree,t}, \boldsymbol{H}^{all,t}\}$.

3.2.1 Local Interaction

The motivation of local communication is to encourage sequential encoder and tree encoder to learn more from each other's pattern of information propagation. In particular, considering sequential encoder $\boldsymbol{H}^{seq,t}$ and tree encoder $\boldsymbol{H}^{tree,t}$ in $\boldsymbol{U}_t$, and current word w_i, as shown in Figure 3. The main idea is to let the nodes in unit $\boldsymbol{U}_t$ take their neighbor nodes of both sequential and tree encoder as input at the last time step $t-1$, including the nodes in sequential model: $h_{i-1}^{seq,t-1}, h_i^{seq,t-1}, h_{i+1}^{seq,t-1}$, and in tree model: $h_{left}^{tree,t-1}, h_j^{tree,t-1}, h_{right}^{tree,t-1}, h_{par}^{tree,t-1}$ (*par* is *parent*), which all are packed into a set $\boldsymbol{H}^{nbs,t-1}$ (*nbs* means *neighbors*).

First, each node in sequential encoder at current step t takes as an additional input the neighbor nodes from the last time step:

$$x_i^{seq,t} = [x_i^{seq}; \overline{x}_i^{seq,t}] \tag{19}$$

where $\overline{x}_i^{seq,t}$ is the neighbor node representation obtained via the attentive operation:

$$u_q^{seq} = v^{\mathrm{T}} \tanh(\boldsymbol{W}_1 \overline{h}_q^{nbs,t-1} + \boldsymbol{W}_2 h_i^{seq,t-1}) \tag{20}$$

$$\alpha_q^{seq} = \mathrm{softmax}(u_q^{seq}) \tag{21}$$

$$\overline{x}_i^{seq,t} = \sum_q \alpha_q^{seq} \overline{h}_q^{nbs,t-1} \tag{22}$$

where $\overline{h}_q^{nbs,t-1} \in \boldsymbol{H}^{nbs,t-1}$ excluding $h_i^{seq,t-1}$ itself.

Similarly, each node in tree encoder takes the additional neighbor node representation as input:

$$x_j^{tree,t} = [x_j^{tree}; \overline{x}_j^{tree,t}] \tag{23}$$

where $\overline{x}_j^{tree,t}$ is formulated as:

$$u_q^{tree} = v^{\mathrm{T}} \tanh(\boldsymbol{W}_1 \overline{h}_q^{nbs,t-1} + \boldsymbol{W}_2 h_j^{tree,t-1}) \tag{24}$$

$$\alpha_q^{tree} = \mathrm{softmax}(u_q^{tree}) \tag{25}$$

$$\overline{x}_j^{tree,t} = \sum_q \alpha_q^{tree} \overline{h}_q^{nbs,t-1} \tag{26}$$

where $\overline{h}_q^{nbs,t-1} \in \boldsymbol{H}^{nbs,t-1}$ excluding $h_j^{tree,t-1}$ itself.

3.2.2 Sentence-level Global Propagation

During sentence-level propagation across recurrent steps T, information exchange between syntax and semantic in a sentence can be extended sufficiently and broadly, and information flow between consecutive words can be enhanced by capturing long-range dependencies.

We reach the goal by employing a context gate over the final node representation $h_i^{all,t}$. Formally,

$$c_i^t = \sigma(W_i^1 h_i^{all,t-1} + W_i^2 \overline{h}_i^{all,t} + b) \tag{27}$$

$$h_i^{all,t} = c_i^t \odot h_i^{all,t-1} + (1 - c_i^t) \odot \overline{h}_i^{all,t} \tag{28}$$

where $\overline{h}_i^{all,t}$ is the ungated value from the concatenation of $h_i^{seq,t}$ and $h_i^{tree,t}$. The context gate c_i^t for the node w_i controls the contribution proportion of history representation and current representation during each step t.

3.3 Decoding and Training

We use a softmax classifier as the decoding layer:

$$y = \mathrm{softmax}(\boldsymbol{r}) \tag{29}$$

Technically, for sentence-level classification, the final sentence representation is the attention representation over $h_i^{all,T}$:

$$\varphi_i = \tanh(\boldsymbol{W} h_i^{all,T} + b) \tag{30}$$

$$\alpha_i = \text{softmax}(\varphi_i) \tag{31}$$

$$\boldsymbol{r}_s = \sum_i^n \alpha_i h_i^{all,T} \tag{32}$$

For sentence pair tasks, such as NLI, we make element-wise production, subtraction, addition and concatenation of two separate sentence representations as a whole:

$$\boldsymbol{r}_p = [\boldsymbol{r}_a; \boldsymbol{r}_b; \boldsymbol{r}_a - \boldsymbol{r}_b; \boldsymbol{r}_a + \boldsymbol{r}_b; \boldsymbol{r}_a \odot \boldsymbol{r}_b] \tag{33}$$

where $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$ are the corresponding sentence representations.

For sequence-level classification, we directly make use of the final node representations $\{h_1^{all,T}, \cdots, h_n^{all,T}\}$, followed by a softmax decoder:

$$y_1, \cdots, y_n = \text{softmax}(h_1^{all,T}, \cdots, h_n^{all,T}) \tag{34}$$

The main task cross-entropy loss can be represented as:

$$\mathcal{L} = -\sum_i \hat{y}_i \log y_i + \frac{\lambda}{2} ||\theta||^2, \tag{35}$$

where $\frac{\lambda}{2}||\theta||^2$ is the l_2 regularization term and $\hat{y}_i$ is the ground truth label.

To avoid cold-start training, we first pre-train the standalone sequential encoder and the tree encoder separately, and then take their parameters as the initial states for the framework at step 0, including $\boldsymbol{H}^{seq,0}$ and $\boldsymbol{H}^{tree,0}$. Thereafter, we train the entire framework in total N iterations with early stopping strategy.

4 Experiments

4.1 Experimental Setups

Hyperparameters. For BiLSTM, TreeLSTM and GCN, we all use a 2-layer version. The dimension of word embeddings is set to 300, which is initialized with the pre-trained GloVe embedding (Pennington et al., 2014). All the hidden sizes in neural networks are set to 350. We adopt the Adam optimizer with an initial learning rate in [1e-5, 2e-5, 1e-6], and L2 weight decay of 0.01. We use the mini-batch in [16, 32, 64] based on the tasks, and apply 0.5 dropout ratio for word embeddings. λ is fine-tuned according to specific tasks.

Task, Dataset and Evaluation. We conduct experiments on typical syntax-dependent tasks. 1) `EFP`, event factuality prediction on the UW dataset (Lee et al., 2015). EFP evaluates the performance of different methods with Pearson correlation coefficient (**r**). 2) `Rel`, relation classification for drug-drug interaction (Segura Bedmar et al., 2013). 3) `SRL`, semantic role labeling on the CoNLL08 WSJ dataset (Surdeanu et al., 2008). Rel and SRL use the F1 score to measure the performance of different models. 4) `NLI`, natural language inference, which also can be modeled as a sentence pair classification, and we investigate NLI on three benchmarks: QNLI (Rajpurkar et al., 2016), SICK (Marelli et al., 2014) and RTE (Bentivogli et al., 2009). For NLI, we use the accuracy to evaluate different models by following previous work.

Note that each dataset contains its own training set, development set and test set. We test the performance of our method 30 times on the corresponding test set, and the results are presented after significant test with p$\leq$0.01. We use the state-of-the-art BiAffine parser (Dozat and Manning, 2017) to obtain the dependency annotation x^{tree}. Being trained on the Penn Treebank (Marcus et al., 1993), the dependency parser has a 93.4% LAS and 95.2% UAS on WSJ test sets.

Baselines. To show the effectiveness of our model, we compare the proposed model with three types of baseline systems.

- Sequential semantic models, including BiLSTM, attention-based BiLSTM, Transformer and sentence-state LSTM (S-LSTM) (Zhang et al., 2018).

- Syntactic tree models, including standalone TreeLSTM or GCN encoder introduced in § 3.1.2.

- Syntax and semantics ensemble models, including ensembling learning (Wolpert, 1992; Ju et al., 2019) and multi-task method (MTL) (Liu et al., 2016).

For ensemble models, we concatenate the output representations of tree encoder TreeLSTM and sequential model BiLSTM. For MTL, we use the underlying shared structure for parameter sharing for TreeLSTM and BiLSTM. For the NLI task, we additionally compare the syntax-semantics integration models, including ESIM (Chen et al., 2017),

System	EFP	Rel	SRL	Avg.
• **Sequential Semantic Model**				
BiLSTM	71.6	80.5	80.0	77.4
AttBiLSTM	73.8	84.9	81.4	80.0
Transformer	74.5	86.2	82.5	81.1
S-LSTM	74.9	86.7	83.0	81.5
• **Syntactic Tree Model**				
TreeLSTM	77.1	87.4	83.3	82.6
GCN	78.5	88.7	83.8	83.7
• **Syntax+Semantics Model**				
Ensemble	76.8	88.1	83.6	82.8
MTL	78.2	88.5	84.0	83.6
• **Ours**				
+TreeLSTM	**80.9***	90.3	**85.7***	**85.6**
+GCN	80.2	**91.2***	84.8	85.2

Table 1: Main results on various tasks. $*$ indicates $p \leq$0.05.

System	QNLI	SICK	RTE	Avg.
• **Sequential Semantic Model**				
BiLSTM	78.6	80.0	58.2	72.3
AttBiLSTM	80.5	81.2	59.6	73.8
Transformer	81.6	82.4	61.8	75.3
S-LSTM	83.9	83.0	63.2	76.7
• **Syntactic Tree Model**				
TreeLSTM	85.2	84.8	66.0	78.7
GCN	83.9	83.7	65.3	77.9
• **Syntax+Semantics Model**				
Ensemble	84.6	84.1	65.0	77.9
ESIM	85.4	85.0	66.7	78.9
StructAlign	86.0	85.3	67.2	80.2
PWIM	86.6	86.1	68.3	79.4
• **Ours**				
+TreeLSTM	**88.2***	**87.2***	**70.6***	**82.0**
+GCN	87.8	86.7	69.0	81.2

Table 2: Results on natural language inference tasks.

StructAlign (Liu et al., 2018) and PWIM (Liu et al., 2019).

4.2 Experimental Results

Main tasks. Table 1 shows the results of different models on EFP, Rel and SRL tasks. Several observations can be found. First of all, the attention-based sequential models (e.g., AttBiLSTM and Transformer) are better than the vanilla BiLSTM model, while the S-LSTM model that incorporates both word-level and sentence-level propagation is more effective in encoding texts, compared with the attention-based sequential models such as AttBiLSTM and Transformer. Besides, tree models with syntactic structure achieve better performance than sequential semantic models, showing the effectiveness of utilizing external syntactic knowledge for syntax-dependent tasks. In particular, the GCN encoder slightly outperforms the TreeLSTM encoder.

In addition, when integrating tree models with sequential networks via ensemble method or multi-task learning, the improvements are quite incremental and limited. Even ensemble learning can be worse than standalone tree encoders such as GCN. Finally, our proposed method (including both the TreeLSTM and GCN based encoder) gives the best results ($p \leq 0.01$) than all the baselines, demonstrating the importance of an effective integration between syntax and semantics. The results also show the TreeLSTM based tree encoder is more beneficial to our deep syntax-semantics communication model. The possible reason is that TreeLSTM encodes syntactic tree structure in an incremental process, during which more detailed information pass can be leveraged. While in GCN, the nodes of syntactic graph is encoded in parallel, though being more computational efficient, offering collapsed information.

NLI tasks. We evaluate different methods on the NLI datasets. As shown in Table 2, similar observations can be found as previous tasks. Among syntactic tree models, TreeLSTM is more effective than GCN for sentences pair encoding, showing the same trends in Table 1. Despite structural architecture of the TreeLSTM encoder, it learns the syntax consecutively, during which more contextual information can be maintained. While GCN encodes the sentence in one shot, it is not sufficient for matching a sentence pair. In addition, we can find that the strong NLI baselines (ESIM, StructAlign and PWIM) give better results than the syntax-semantics ensemble model, as they can provide more sophisticated manner on incorporating syntactic knowledge with semantic composition. Nevertheless, our model outperforms all baseline systems, with an average accuracy 82.0% by the TreeLSTM tree baseline, and 81.2% by the GCN encoder. The above results prove that our framework is highly effective in integrating syntactic structure with sequential semantic models.

4.3 Ablation Study

We do ablation tests to analyze the contribution of different components, including tree encoders,

Item	Acc.
Sequential Sem. (+TreeLSTM)	
+Xavier	87.3
+ELMo	89.4
+BERT	92.8
Communication (+TreeLSTM)	
w/o Local (Eq. 23 & 19)	85.1
w/o Global$_a$ (Eq. 28)	86.9
w/o Global$_b$ (T=0)	86.4

Table 3: Ablation study on QNLI.

sequential encoders and communication methods. The experiments are based on the QNLI dataset. As shown in Table 1 and 2, TreeLSTM performs better than GCN on NLI tasks. But the result is the opposite on the other tasks, as we discussed earlier. This shows that different syntax-dependent tasks rely on different tree encoders with differing utility. We further investigate the influences of sequential encoders, as shown in Table 3. First, we replace the GloVe embedding with the one initialized by the Xavier algorithm (Glorot and Bengio, 2010), and we can find that the performance has a significant drop. When we use the state-of-the-art language models, such as ELMo and BERT, instead of BiLSTM, we obtain prominent performance gains. This indicates the importance of semantics for the framework. Second, if we abandon the local communication mechanism, the accuracy decreases dramatically. Finally, the context gate c_i^t and the sentence-level propagation architecture make similar contributions on global communication for the task performance.

4.4 Efficiency

We investigate the efficiency of different models on the EFP task. As shown in Figure 4, our method gives competitive performance when the sentence length grows to 30. In contrast, the performance of the other models has been significantly reduced. This indicates that our method can partially relieve the long distance dependency problem, thanks to the sentence-level global communication.

We explore the impact of recurrent steps in sentence-level propagation architecture. As shown in Figure 5, TreeLSTM converges at step 7, while GCN is faster, at step 3. This partially coincides with the principle that GCN is more efficiency-saving than TreeLSTM.

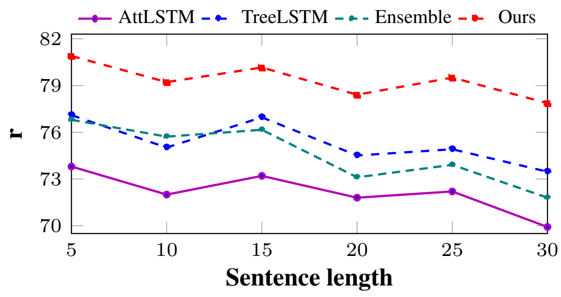


Figure 4: Performance of different sentence lengths.

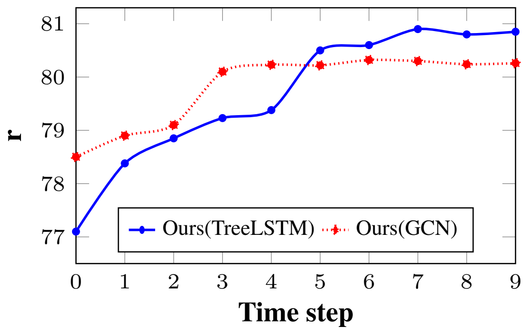


Figure 5: Sentence-level propagation under varying time step.

4.5 Semantic Composition

We explore to what extent the model better captures the semantics of sentences. Based on the RTE task, we first measure the distance of each sentence pair on semantic representation with the **Euclidean** distance (Ed), and then scale the continuous value $\hat{y}_i$ into [0,1]. The RTE gold test labels $y_i \in \{0, 1\}$, includes *Entailment* and *Contradict*. We define a semantic deviation as: $Dev(y, \hat{y}) = \sqrt{\frac{1}{N}\sum_i^N (Ed(y_i, \hat{y}_i) - \overline{Ed})}$, where $\overline{Ed}$ is the averaged distance. If all the predicted distances are coincident with gold ones, or different from oracles, Dev=0, indicating the maximum consistency of semantic representation. We make statistics for the deviation of each sentence pair by several baselines, as shown in Figure 6. We can see that our method gives the best semantic consistencies with gold ones, compared with other methods. This indicates the effectiveness of our model on composing sentence semantics.

We also explore the effectiveness of semantic composition by comparing our method with the NLI model StructAlign. We scatter the predicted probability of each sentence pair. Technically, a model is expected to predict the *Entailment* label (y=1) with larger probability, and vise versa for the *Contradict* label. As shown in Figure 7, we

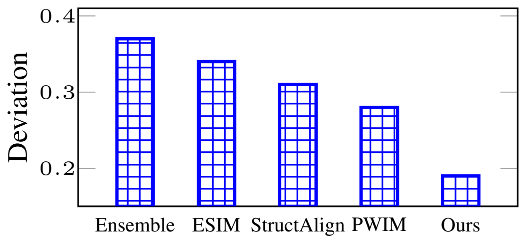


Figure 6: Semantic deviation of sentence pair.

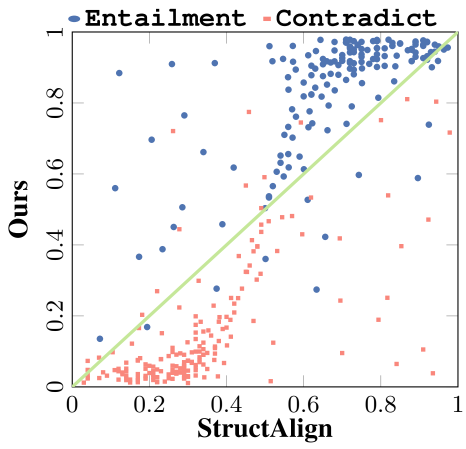


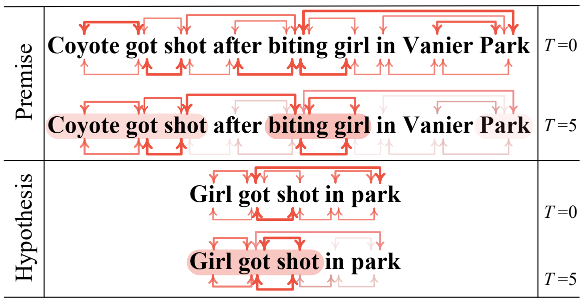
Figure 7: Scatter plot on accuracy probability for RTE.

find that our model tends to predict the samples with higher confidence, on either *Entailment* or *Contradict*, compared with StructAlign. The above analysis proves that our method is more effective than StructAlign in semantics composition.

4.6 Case Study

To better understand the introduced local and global communication mechanisms, we conduct case study by empirically visualizing the edge weights on both syntactic dependency structure encoder and sequential model, and the word weights. The experiment is based on the RTE test set. We first calculate the attention weights for each node in sequential model (Eq. 21) and tree model (Eq. 25), respectively. We then recompute such edge weights for all the nodes via global normalization at sentence level, for sequential model and tree model, separately. By calculating the co-occurrence matrix of edge weights, we can obtain word weights. We visualize the importance of words, dependent edges and consecutive edges for *Premise* and *Hypothesis* sentences, respectively.

As illustrated in Figure 8, before information exchange (T=0), the weights of dependency edges and consecutive edges are inaccurate and not directly useful for capturing semantics. Besides, the 'attentions' focused on edges in sequential model and tree model are quite different. When the framework is trained close to convergence, at time step 5, the connections between syntax and semantics tend to be mutually coincident. The possible reason is that sequential semantics can guide syntactic structure learning onto the proper place. For example, syntactic links for ***biting girl*** in Premise, and the one for ***got shot*** in Hypothesis, are enhanced by the correspondences of sequential edges, respectively. Consequently, more informative words, e.g., ***biting girl*** in Premise and ***Girl got shot*** in Hypothesis, can receive proper weights for building more accurate semantics. With such semantics composition, the model easily gives correct predictions. This shows that an effective communication can improve mutual learning of syntactic structure and sequential semantic.



Premise
Coyote got shot after biting girl in Vanier Park
T=0
Coyote got shot after biting girl in Vanier Park
T=5
Hypothesis
Girl got shot in park
T=0
Girl got shot in park
T=5

Figure 8: Case visualization of words and edges on the RTE task (with *Contradict* label). The arrows above sentences are bidirectional syntactic dependents, and the ones below sentences are sequential semantics.

5 Conclusion

We proposed a deep syntax-semantics communication model for improving text understanding. Local communication was performed between syntactic tree encoder and sequential semantic encoder for mutual learning of information exchange. Global communication was performed for ensuring information propagation throughout entire architecture over recurrent steps. Results on multiple tasks showed that our model outperformed strong baselines. In-depth analysis further indicated that our method was highly effective on composing sentence semantics.

Acknowledgments

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the ICLR*.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL*.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the NAACL*, pages 2453–2464.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the ACL*, pages 1657–1668.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the EMNLP*, pages 1724–1734.

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*.

Hao Fei, Yafeng Ren, and Donghong Ji. 2020a. Negation and speculation scope detection using recursive neural conditional random fields. *Neurocomputing*, 374:22–29.

Hao Fei, Yafeng Ren, and Donghong Ji. 2020b. A tree-based neural network model for biomedical event trigger detection. *Information Science*, 512:175–185.

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

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the AISTATS*, pages 249–256.

Serhii Havrylov, Germán Kruszewski, and Armand Joulin. 2019. Cooperative learning of disjoint syntax and semantics. In *Proceedings of the NAACL*, pages 1118–1128.

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

Ying Ju, Fubang Zhao, Shijie Chen, Bowen Zheng, Xuefeng Yang, and Yunfeng Liu. 2019. Technical report on conversational question answering. *CoRR*, abs/1909.10772.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the ICLR*.

Kenton Lee, Yoav Artzi, Yejin Choi, and Luke Zettlemoyer. 2015. Event detection and factuality assessment with non-expert supervision. In *Proceedings of the EMNLP*, pages 1643–1648.

Linqing Liu, Wei Yang, Jinfeng Rao, Raphael Tang, and Jimmy Lin. 2019. Incorporating contextual and syntactic structures improves semantic similarity modeling. In *Proceedings of the EMNLP*, pages 1204–1209.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the IJCAI*, pages 2873–2879.

Yang Liu, Matt Gardner, and Mirella Lapata. 2018. Structured alignment networks for matching sentences. In *Proceedings of the EMNLP*, pages 1554–1564.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. 2015. A dependency-based neural network for relation classification. In *Proceedings of the ACL*, pages 285–290.

Moshe Looks, Marcello Herreshoff, DeLesley Hutchins, and Peter Norvig. 2017. Deep learning with dynamic computation graphs. In *Proceedings of ICLR*.

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.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the LREC*, pages 216–223.

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 *Proceedings of the NAACL*, pages 2227–2237.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the EMNLP*, pages 2383–2392.

Isabel Segura Bedmar, Paloma Martínez, and María Herrero Zazo. 2013. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts. In *Proceedings of the SemEval*, pages 341–350.

Peng Shi, Zhiyang Teng, and Yue Zhang. 2016. Exploiting mutual benefits between syntax and semantic roles using neural network. In *Proceedings of the EMNLP*, pages 968–974.

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.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Proceedings of the INTERSPEECH*, pages 194–197.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the CoNLL*, pages 159–177.

Swabha Swayamdipta, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Greedy, joint syntactic-semantic parsing with stack LSTMs. In *Proceedings of the CoNLL*, pages 187–197.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of the EMNLP*, pages 3772–3782.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Van-Hien Tran, Van-Thuy Phi, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Relation classification using segment-level attention-based CNN and dependency-based RNN. In *Proceedings of the NAACL*, pages 2793–2798.

Shikhar Vashishth, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya, and Partha Talukdar. 2019. Incorporating syntactic and semantic information in word embeddings using graph convolutional networks. In *Proceedings of the ACL*, pages 3308–3318.

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 *Proceedings of the NIPS*, pages 5998–6008.

Yufei Wang, Mark Johnson, Stephen Wan, Yifang Sun, and Wei Wang. 2019. How to best use syntax in semantic role labelling. In *Proceedings of the ACL*, pages 5338–5343.

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.

Xingdi Yuan, Marc-Alexandre Côté, Jie Fu, Zhouhan Lin, Chris Pal, Yoshua Bengio, and Adam Trischler. 2019. Interactive language learning by question answering. In *Proceedings of the EMNLP*, pages 2796–2813.

Yuan Zhang and Yue Zhang. 2019. Tree communication models for sentiment analysis. In *Proceedings of the ACL*, pages 3518–3527.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state LSTM for text representation. In *Proceedings of the ACL*, pages 317–327.