PHRASE-LEVEL GLOBAL-LOCAL HYBRID MODEL FOR SENTENCE EMBEDDING

Mingyu Tang∗†, Liansheng Zhuang∗†, Houqiang Li∗, Jian Yang†‡, Yanqun Guo§

∗University of Science and Technology of China, †Peng Cheng Laboratory
‡Northern Institute of Electronic Equipment, §Southwest Jiaotong University
tmy528@mail.ustc.edu.cn, lszhuang@ustc.edu.cn

ABSTRACT

Latent structure models have drawn much attention due to the ability to learn an optimal latent hierarchical structures without explicit structure annotations. However, most existing models suffer from high computation complexity and hard training. To this end, this paper proposes a novel phrase-level global-local hybrid model, which inherits the advantages of existing latent structure models while requires less time complexity. Our model splits a sentence into multiple phrases by a category-selection module. Then, it encodes the context dependency by a phrase-level global encoding module, and encodes the task-specific information by a phrase-level local encoding module. Finally, sentence embedding is obtained by integrating the global encoding and task-specific encoding. Experiments on public benchmarks show that, our model achieves state-of-the-art performance on the tasks of sentence classification and natural language inference. Meanwhile, our model is at least 10 times faster than existing state-of-the-art method at the training stage.

Index Terms— sentence embedding, latent structure model, phrase level

1. INTRODUCTION

Sentence embedding plays a critical role in many natural language processing (NLP) applications such as sentiment analysis [1], question answering [2] and entailment recognition [3]. It aims at mapping sentences into dense real-valued vectors that represent their semantics. Though word embedding has achieved great success [4], there is no clear way to build embedding that carries full meaning of a sentence, which is the basic processed unit in most downstream tasks.

Previous methods for sentence embedding mainly rely on Recurrent Neural Networks (RNNs) to generate context-aware representation [5]. Though RNNs encoders are capable of learning long-term dependencies by reading words in sequential order, they are hard to parallelize and not time-efficient.

Beyond that, attention-based methods [6] use attention mechanism to build representations by scoring input words differentially. Beyond that, attention-based methods [6] use attention mechanism to build representations by scoring input words differentially, which can be more parallelizable and requires significantly less time to train than RNNs. Both RNNs models and attention-based models have achieved striking performance, but they can not well encode linguistic information of natural language which can impact their perform to some extent. Language is inherently tree structured, and the meaning of a sentence comes largely from composing the meanings of sub-trees. To explicitly model the composition, tree-based models [7, 8] are proposed to use pre-specified parsing trees to embed sentences by recursive manner. Although there is significant benefit in processing a sentence in a tree-structured, data annotated with parse trees could be expensive to prepare and hard to be computed in batches which seriously affect training efficiency.

Recently, latent tree models have been proposed to learn the optimal hierarchical latent structure of text from sequence into sentence representation without specific tree annotation [9]. The training signals to parse and embed sentences are both from certain downstream tasks. However, most existing latent structure models suffer from time consuming problems and hard training due to the ineffective reinforcement learning method [10] or heavy calculation for candidate parent tree nodes [11]. Meanwhile, most of these models still follow the way of parsing sentence to binary tree with words in leaf nodes and then composing adjacent node pairs bottom up. This prevents the sentence embedding from focusing on the most informative words, resulting in a performance limitation on certain tasks [12]. We argue that weak form of syntax tree can be better for sentence embedding by neural network.

Motivated by above insights, this paper proposes a novel phrase-level global-local hybrid (PLGLH) model, which inherits the advantages of existing latent structure models with less time complexity. Our model splits a sentence into multiple phrases by a category-selection module rather than reinforce learning method like HS-LSTM [11], and obtains a weak form of constituent in the syntax of a sentence. To filter out information that is semantically or syntactically distant, each phrase is encoded respectively. To capture the context

978-1-7281-1331-9/20/$31.00 ©2020 IEEE
dependencies among phrases, a phrase-level global encoding module is introduced based on LSTM, and process these phrases by sequence. Since different phrases have different relevance to the task, our model introduce a phrase-level local encoding module based on attention mechanism to emphasize the task-informative phrases. The final sentence embedding is obtained by fusing the global encoding and local encoding, and then applied in a downstream application. Compared with existing latent tree models which are based on reinforcement learning or recursive composing, our model is easy to train, and significantly reduces the training time.

We evaluate our model’s performance on a plethora of datasets of natural language inference and sentence classification. Experiment results show that our model outperforms or is at least comparable to previous sentence encoder models.

The main contributions of our work can be summarized as follows:

- we propose a novel phrase-level global-local hybrid model for sentence encoding that can combines global context dependency and task-specific local information.
- Our model inherits the advantages of existing latent structure with less time complexity.
- Our proposed model outperforms the state-of-the-art supervised sentence embedding method on a wide range of datasets.

2. PROPOSED MODEL

In this section, we introduce the PLGLH model for sentence encoding. Formally, given a sentence $X = (x_1, x_2, \ldots, x_T)$ with length of $T$, where $x_t$ represents the word embedding of corresponding word, our aims is to build a fix size vector $emb$ to summarize the information of it. The overall process is shown in Figure 1.

2.1. Overview

First, our model applies a word-level module to split a sentence into multiple phrases by Straight-Trough (ST) Gumbel-Softmax estimator [13] rather than similar HS-LSTM which uses reinforcement learning method, and obtains a weak form of constituent in the syntax of a sentence. To filter out information that is semantically or syntactically distant, each phrase is encoded respectively, detail in section 2.2. To capture the context dependencies among phrases, a phrase-level global encoding module is introduced based on LSTM, and process these phrases by sequence, detail in section 2.3. Since different phrases have different relevance to the task, our model introduce a phrase-level local encoding module based on attention mechanism to emphasize the task-informative phrases, detail in section 2.4. The final sentence embedding is obtained by fusing the global encoding and local encoding.

2.2. Word-level Phrase Division

Our model splits sentence into phrases and build representation for them in this phase. In particular, it is formulated as a sequence decision by action $a_t \in \{\text{Inside}, \text{End}\}$. When $a_t$ is $\text{Inside}$ means $x_t$ is the start of one phrase, or belongs to same phrase as $x_{t-1}$, otherwise $a_t$ is $\text{End}$ means $x_t$ is the end of that phrase. As result, words in sentence will be split into a few of consecutive parts, or called phrase.

We use a word-level LSTM to connect a sequence of words to form a phrase representation. The transition of the word-level LSTM depends upon action $a_{t-1}$. If action $a_{t-1}$ is $\text{End}$, the word at position $t$ is the start of a phrase and the word-level LSTM start with a initialized state, which also can be trained in the training phase. Otherwise the action is $\text{Inside}$ and the world-level LSTM continues from its previous states. The process is described formally as follows:

$$e_t^w, h_t^w = \begin{cases} \Phi_w^w(x_t, e_{t-1}^w, h_{t-1}^w), & a_{t-1} = \text{End} \\ \Phi_w^w(x_t, e_{t-1}^w, h_{t-1}^w), & a_{t-1} = \text{Inside} \end{cases}$$

where $\Phi_w^w$ denotes the transition function of word-level
LSTM, \( c_t \) is the memory cell, and \( h_t \) is the hidden state at timestep \( t \).

Since the action of phrase division is not given to the model, we need to build up the structure at the same time. We choose the ST Gumbel-Softmax estimator [13] to achieve sampling from discrete action space. It is a method of utilizing discrete random variables in a network by approximating sampling from discrete action space. We choose the ST Gumbel-Softmax estimator \([13]\) to achieve model, we need to build up the structure at the same time.

Let \( c \) be a \( k \)-dimensional categorical distribution whose class probabilities \( \pi_1, \pi_2, \ldots, \pi_k \) are defined by unnormalized log probabilities \( \pi_1, \pi_2, \ldots, \pi_k \), as show in equation 2. Then ST Gumbel-Softmax is processed as equations 3.

\[
p_i = \frac{\exp(\log(\pi_i))}{\sum_{j=1}^{k} \exp(\log(\pi_j))}
\]

\[
u_i \sim \text{Uniform}(0, 1)
\]

\[g_i = -\log(-\log(u_i))\]

\[y_i = \frac{\exp(\log(\pi_i) + g_i)/\tau)}{\sum_{j=1}^{k} \exp((\log(\pi_j) + g_j)/\tau)}\]

\[y_i^{ST} = \begin{cases} 1, & i = \arg\max_j y_j \\ 0, & \text{otherwise} \end{cases}\]

where \( u_i \) is random noise; \( \tau \) is temperature parameter and be set to 1 in our experiment; \( y = (y_1, y_2, \ldots, y_k) \) is similar as standard softmax output and can be used for backpropagation; \( y^{ST} = (y_1^{ST}, y_2^{ST}, \ldots, y_k^{ST}) \) is one-hot vector that can represent discrete sampling in forward propagation.

Come back to our model, we use linear transformation to map the word embedding and hidden state to two-dimensional action space, represented as \( (\pi_1, \pi_2) \), and then use ST Gumbel-Softmax to sample the action from it at every time step.

The output \( a_t \) at forward propagation will be \((1, 0)\) or \((0, 1)\), representing the action of End or Inside.

Finally we will get a sequence of outputs of hidden states and actions, \((h_1, h_2, \ldots, h_T)\) and \((a_1, a_2, \ldots, a_T)\), where \( T \) is the length of input words. When \( a_t = (1, 0) \), it indicates that \( h_t \) represents the meaning of some phrase. Otherwise, \( h_t \) is invalid and won’t be used in later. But in order to support batched computation, we won’t throw away the invalid part of \( h \) directly.

After that, each phrase has been encoded, and we can use them to build the embedding for sentence.

### 2.3. Phrase-level Global Encoding

Follow the common practice in latent structure model, we use another phrase-level LSTM to capture the context dependency of whole sentence from sequence of phrases.

As we can see in Figure 1, actions produced in word-level module will guide us to select valid hidden state. When the action \( a_t \) is End, a phrase ends at position \( t \) and the hidden state \( h_t \) which carrying means of it will be fed into phrase-level LSTM. Otherwise the action is Inside and the phrase-level LSTM is fixed at this step, and the variables are copied from the preceding position. Formally as follow:

\[
e_{t_p}^P, h_t^P = \begin{cases} \Phi^P(h_{t_p}^w, r_{t-1}^P, h_{t-1}^P), & a_{t-1} = \text{End} \\ e_{t_p}^P, h_t^P, & a_{t-1} = \text{Inside} \end{cases}
\]

where \( \Phi^w \) denotes the transitions function of phrase-level LSTM, \( h_t^w \) is the output of word-level LSTM. The last output hidden state of phrase-level LSTM will be considered as the global encoding for the sentence.

### 2.4. Phrase-level Local Encoding

Since different phrases have different relevance to specific task, our model introduce a phrase-level local encoding module based on attention mechanism to emphasize the task-informative phrases. We follow the Scaled Dot-Product Attention of Transformer [14].

First, filter the invalid hidden state. Different with global encoding with LSTM, we can just force the invalid item to zero. Then map each valid hidden state to query(Q), key(Key) and value(V) simultaneously. Inner product of key and query can indicate the relevance between two phrases. New summarize representation for each time step is the weighted sum of each part’s value which simulates the relevance as each part can exchange information directly and help to find the important message. The whole process can be seen in Fig 2, formally in matrix form as follows:

\[
\hat{h}^w = \text{ActionMask}(h^w, a)
\]

\[
Q, K, V = \text{Linear}(\hat{h}^w)
\]

\[
Z = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V
\]

\[
Z_{out} = \text{MeanPooling}(Z)
\]

where \( \text{ActionMask} \), as the name, use the actions generated in word-level module to mask the \( h^w \) to force the invalid parts to zero. \( \text{Linear} \) denotes the linear transformation without bias, which keeps the zero items still be zero and transfers hidden state to query, key and value vectors separately. \( d_k \), which denote the dimension of Q and K, is used as scaling factor. Mean-pooling is used to map the attention module output to fix size embedding.

Finally, we can fusion two embeddings to get the sentence representation compressed with local and global information. In this paper, we just simply concatenate two embeddings as the final result to keep information as much as possible.

\[
\text{emb} = h_T^P \oplus Z_{out}
\]
### 3. EXPERIMENTS AND ANALYSIS

In this section, we conduct a plethora of experiments to study the effectiveness of our model, including several datasets in classification and language inference. In order to study the influence of each module, we conduct module analysis experiments by deleting global encoding module or local encoding module. PLGLH\_global refers to the model without global encoding. PLGLH\_local refers to the model without local encoding. In all of our experiments, 300-dimensional GloVe [4] word embeddings are used to represent words, out-of-vocabulary words are mapped to zero vectors with same size like others. All the word embeddings retain fixed during training. Minibatch size is set to 64. To train the model, Adam optimizer with learning rate 1e-3 is used. The drop out rate is 0.2.

#### 3.1. Language Inference Task

Natural language inference(NLI) is a fundamental task in the field of NLP that involves reasoning about semantic relationship between sentences. So that the performance of NLI is a very important measurement for sentence embedding quality.

In this experiment, we choose two datasets including Stanford Natural Language Inference(SNLI) dataset [3] and Multi-Genre Natural Language Inference (MultiNLI) dataset [20]. SNLI dataset has 550k human-annotated sentence pairs, called premise and hypothesis, each labeled with one of the following pre-defined relationship: Entailment, means the premise entails the hypothesis; Contradiction, means they contradict with each other; Neutral, means they are irrelevant. MultiNLI is almost same form like SNLI, but covers a range of genres of spoken and written text and supports cross-genre evaluation. There is some examples of them in Table 1.

Following the previous work, we use a siamese architecture that applying the model to both premise and hypothesis, $emb_p$ and $emb_h$ are fixed-length vector representations for premise and hypothesis respectively. The final sentence-pair representation is formed by concatenating the original vectors with the absolute difference and element-wise multiplication between them:

$$emb^{\text{inp}} = [emb^p; emb^h; |emb^p - emb^h|; emb^p \odot emb^h]$$

At last, we feed the sentence-pair representation $emb^{\text{inp}}$ into

#### Table 1. Dataset examples

<table>
<thead>
<tr>
<th>name</th>
<th>N</th>
<th>example</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNLI</td>
<td>550k</td>
<td>Two women are embracing while holding to go packages. (premise) Two woman are holding packages. (hypothesis)</td>
<td>entailment</td>
</tr>
<tr>
<td>MultiNLI</td>
<td>433k</td>
<td>I don’t know um do you do a lot of camping. (premise) I know exactly. (hypothesis)</td>
<td>contradiction</td>
</tr>
<tr>
<td>MR</td>
<td>11k</td>
<td>Unfortunately the story and actors are served with a hack script. This is by far the nicest one, in so many ways.</td>
<td>neg</td>
</tr>
<tr>
<td>CR</td>
<td>4k</td>
<td>Smart and alert, thirteen conversations about one thing is a small gem.</td>
<td>pos</td>
</tr>
<tr>
<td>SUBJ</td>
<td>10k</td>
<td>Smart and alert, thirteen conversations about one thing is a small gem.</td>
<td>sub</td>
</tr>
</tbody>
</table>

#### Table 2. NLI experiment results. Evaluation metric is the classification accuracy in test for SNLI and in matched evaluation for MultiNLI. Result of previous model is come from the paper where the model is proposed if not point out specially.

<table>
<thead>
<tr>
<th>Models</th>
<th>SNLI</th>
<th>MultiNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bi-LSTM+Maxpooling [5]</td>
<td>84.5</td>
<td>68.2 [15]</td>
</tr>
<tr>
<td>shortcut-stacked BiLSTM [16]</td>
<td>86.1</td>
<td>-</td>
</tr>
<tr>
<td>DiSAN [17]</td>
<td>85.6</td>
<td>-</td>
</tr>
<tr>
<td>Reinforced self-attention [18]</td>
<td>86.3</td>
<td>[19] -</td>
</tr>
<tr>
<td>TC-RN [19]</td>
<td>85.7</td>
<td>-</td>
</tr>
<tr>
<td>SPINN [9]</td>
<td>82.6</td>
<td>68.2 [15]</td>
</tr>
<tr>
<td>PLGLH_global</td>
<td>86.4</td>
<td>72.6</td>
</tr>
<tr>
<td>PLGLH_local</td>
<td>86.1</td>
<td>71.4</td>
</tr>
</tbody>
</table>

![Fig. 2. Illustration of self-attention module.](image)
a classification network with two layers of linear transformation. The hidden dimension of linear transformation is 512. This is the standard scheme for sentence encoders trained on SNLI.

Experiment results are listed in Table 2. Evaluation metric is the classification accuracy in test for SNLI and in mismatched evaluation set 1 for MultiSNLI. Result in SNLI shows that our model outperform the sequence models like the BiLSTM with maxpooling [5] and shortcut-stacked BiLSTM [16], attention based model like DiSAN [17], get better result than Reinforced-selfattention [18], which use REINFORCE algorithm. Comparment with other latent tree model like SPINN [9] and gumbel Tree-LSTM [11] or model use tree structure as constrain like TC-RN [19], our model also give a better accuracy. When comes to the MultiNLI dataset, there is not so much works about evaluation on this single dataset, we just compare with the baseline model BiLSTM and some latent tree model like SPINN and gumbel Tree-LSTM. Results show that our model can still get better performance.

3.2. Classification Task

Sentence representation is widely applied in single sentence classification, and it is the most common task of NLP. We evaluate our model on various datasets as listed bellow. Table 1 also give some examples of them.

- The Customer Reviews(CR) dataset. Dataset consist of reviews for some products, where the task is to classify each customer review as a positive or negative review [21].
- The Movie Reviews(MR) dataset. Dataset consist of reviews sentence for a movie with an assigned positive or negative label about reviewers attitude [1].
- The Subjectivity(SUBJ) dataset is to classify a sentence as being subjective or objective [22].

For each experiment, we apply our model to get the sentence embedding $emb$, and then use a classification network with one layer of linear transformation. The hidden dimension of linear transformation is 100.

Experiment results are listed in Table 3. Our model outperform models such as baseline model BiLSTM and CNN-Ana [23], tree constrained models like DC-TreeLSTM [24] and SATA Tree-LSTM [26] or latent tree model HS-LSTM [10] and gumbel Tree-LSTM [11]. Only TreeNet [8] get better result than ours in one task. But our model requires significantly less time to train. In order to explain intuitively, we record time consuming of other two model: TreeNet, which gets best result but can’t compute in batches; gumbel Tree-LSTM, which also apply ST gumbel-softmax to latent tree model, but suffer from heavy calculation for candidate parent tree nodes. Result shows in Fig 3. Result shows our model is at least 10 times faster than existing state-of-the-art method at the training stage. Note that we doesn’t compare time consuming with reinforce learning method since it obviously slow and hard to convergence [13].

3.3. Module Analysis

Results of module analysis experiments show that performance gets slight decrease in all dataset without local encoding or global encoding module. This phenomenon suggests the significance of combining of global and local encoding. Not only that, result of PLGLH\local is better than

\footnotesize

<table>
<thead>
<tr>
<th>Models</th>
<th>MR</th>
<th>CR</th>
<th>SUBJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>BiLSTM [5]</td>
<td>80.7</td>
<td>83.8</td>
<td>93.4</td>
</tr>
<tr>
<td>CNN-Ana [23]</td>
<td>81.0</td>
<td>84.65</td>
<td>93.67</td>
</tr>
<tr>
<td>DC-TreeLSTM [24]</td>
<td>81.7</td>
<td>-</td>
<td>93.7</td>
</tr>
<tr>
<td>AdaSent [25]</td>
<td>83.1</td>
<td>86.3</td>
<td>95.5</td>
</tr>
<tr>
<td>TreeNet [8]</td>
<td>83.8</td>
<td>88.4</td>
<td>95.9</td>
</tr>
<tr>
<td>gumbel Tree-LSTM [11]</td>
<td>83.0</td>
<td>86.1</td>
<td>94.9</td>
</tr>
<tr>
<td>HS-LSTM [10]</td>
<td>82.1</td>
<td>-</td>
<td>93.7</td>
</tr>
<tr>
<td>SATA Tree-LSTM [26]</td>
<td>83.8</td>
<td>-</td>
<td>95.4</td>
</tr>
<tr>
<td>PLGLH\local</td>
<td>83.0</td>
<td>86.0</td>
<td>94.6</td>
</tr>
<tr>
<td>PLGLH\global</td>
<td>83.7</td>
<td>86.5</td>
<td>95.1</td>
</tr>
<tr>
<td>PLGLH</td>
<td>84.6</td>
<td>87.4</td>
<td>95.9</td>
</tr>
</tbody>
</table>

\footnotesize

To see more detail about MultiNLI in https://www.nyu.edu/projects/bowman/multinli/
PLGLH: global in the natural language inference experiments, when opposite results in classification tasks. It verifies our suppose since natural language inference is determined by whole sentence semantic while sentence classification takes more care of specific parts like verb.

4. CONCLUSION

This paper proposes a PLGLH model for sentence embedding which inherits the advantages of existing latent structure models while requires less time complexity and can combine global context dependency information and task-specific local information at phrase level. From experiment results on a wide range of datasets, we validate the effectiveness of our model. Results empirically show that hierarchical information and global-local is important for understanding sentence and latent tree model is an effective and efficient way to replace recursive tree model.

5. REFERENCES