INTEGRATING DEPENDENCY TREE INTO SELF-ATTENTION FOR SENTENCE REPRESENTATION

Junhua Ma* *Jiajun Li*† *Yuxuan Liu** *Shangbo Zhou** *Xue Li*†

* College of Computer Science, Chongqing University, Chongqing, China
†School of Information Technology & Electrical Engineering,
The University of Queensland, St Lucia, Qld, Australia

ABSTRACT

Recent progress on parse tree encoder for sentence representation learning is notable. However, these works mainly encode tree structures recursively, which is not conducive to parallelization. On the other hand, these works rarely take into account the labels of arcs in dependency trees. To address both issues, we propose Dependency-Transformer, which applies a relation-attention mechanism that works in concert with the self-attention mechanism. This mechanism aims to encode the dependency and the spatial positional relations between nodes in the dependency tree of sentences. By a score-based method, we successfully inject the syntax information without affecting Transformer's parallelizability. Our model outperforms or is comparable to the state-of-the-art methods on four tasks for sentence representation and has obvious advantages in computational efficiency.

Index Terms— Dependency tree, Transformer, Relation, Attention

1. INTRODUCTION

In recent years, distributed sentence representations have been used in many natural language processing(NLP) tasks. Different composition operators have been used to map lexical representations to single sentence representations, such as recurrent neural network(RNN)[1, 2, 3], convolutional Neural Networks(CNN)[4, 5], recursive convolutional neural network(RCNN)[6, 7] and Transformer[8]. Despite their success, these methods fail to explicitly take advantage of syntactic information in sentences.

A parse tree shows the syntactic structure of a sentence and contains grammatical information. To make use of the information, three recursive[9, 10, 11] models were proposed to encode a sentence along with its parse tree, using several compositional functions to integrate tree nodes following a bottom-up manner. [12] proposed two LSTM-based models to encode sentences along with constituency tree and dependency tree recursively. Unfortunately, models based on RNN have a bad parallelizability. The heterogeneity of tree structure inputs makes it impossible to train multiple samples simultaneously in a batch, which further limits the parallelizability of the model.

On the other hand, Transformer has been extremely popular since its introduction because of its superior parallelism and performance. Many research tried to integrate the parse tree information into Transformer. Tree-Transformer[13] recursively encode words with their parents and children in the parse tree by a Transformer module. Although applied the transformer structure, the model still followed a recursive mechanism, which cannot be trained in parallel. Besides, dependency tree structure includes two parts of information: topological structure and types of dependency relations(label of dependency arc). These labels also provide critical information about the dependency tree, but few models can distinguish them well. This raises a question: can we utilize the information included in the syntax tree as much as possible without affecting the original mechanism so that it keeps its parallelizability?

In this paper, we propose a Transformer-based model that applies the relation-attention mechanism, Dependency-Transformer, which provides a positive answer to the question above. The relation-attention mechanism integrates the dependency information into the self-attention mechanism in two steps. Firstly, we set learnable vectors for dependency relations and the relative positions in the tree, then pool them to scores in each attention head. Secondly, to link relation attention scores and the semantics of sentences, we implement a gating mechanism that enables these relation-attention scores to adjust the proportion that will be added into the self-attention scores by the lexical semantics.

We evaluate our model on four widely-used datasets of sentence representation tasks. Our model outperforms all baselines and has obvious advantages over SICK-E and SICK-R. For the SST-2 and MRPC, the model's performance is comparable to state-of-the-art methods. Moreover, our model improves while the efficiency is significantly ahead of the recursive tree model. Furthermore, we conduct an ablation study and case study, which further validate the efficiency and rationality of the proposed mechanism.

978-1-6654-0540-9/22/$31.00 ©2022 IEEE

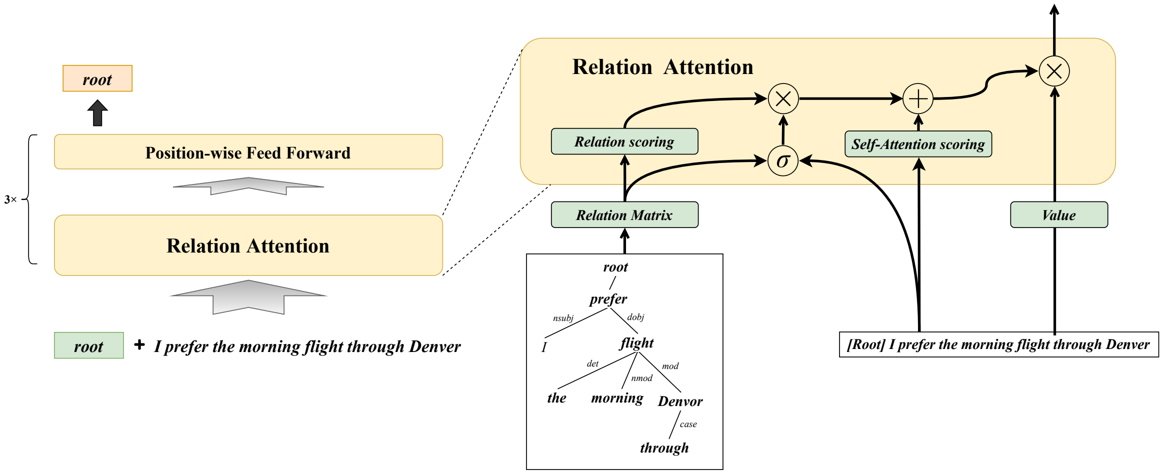


Fig. 1. Overall structure of dependency-Transformer.

2. PROPOSED MODEL

2.1. Model Architecture

Our model's structure, shown in Figure 1, is based on a three-layer Transformer encoder. Given an input sequence $X = [x_1, x_2, ..., x_n] \in \mathbb{R}^{l\times d}$, a $[root]$ token will be inserted at the beginning of the sentence, which represents the $ROOT$ node of the parse tree, and then $X = [root, x_1, x_2, ..., x_n]$. At the end, the model will yield the sequence's output $Y' = [y_{root}, y_0, y_1, .., y_n] \in \mathbb{R}^{(l+1)\times d}$. We take the output of the $ROOT$ node as the representation of the sentence.

The input of Transformer is the sum of two parts: the word embeddings and the positional vectors, which provide the distributional information about words of the training corpus and the words' positional information in the sentence. To enrich the expression of words, we introduce the level embeddings, which are defined as the levels of words in the parse tree (the distance from the root node). Similar to the positional vectors, we set learnable vectors for different words' levels and add them to the above embeddings together as the input of our model.

2.2. Dependency relation

As mentioned earlier, one of the previous works' limitations is the lack of utilization of relation types in the dependency tree. Therefore, our primary concern is how to incorporate these relations into Transformer and give them distinct attention. In self-attention of Transformer, attention scores are scaled-dot products of the hidden states of words, which may represent a kind of correlation between words. They are expected to capture the lexically semantic correlations. Therefore, we can also view the words' relation in the dependency parse tree as a correlation between words and inject it into self-attention as part of the attention scores. Correspondingly, These scores are expected to capture the syntactic information of the sentence. In the implementation, we set a learnable vector for each relation(The dimensions of these vectors are relatively small in consideration of the impact on computational efficiency). Different linear layers are applied at each attention head to calculate scores for these vectors. And then, these scores will be added to the original attention scores as a bias. The relation-attention score($word_i$ attend $word_j$) in a specific attention head is computed as follow(we omit the superscript of layer number for simplicity):

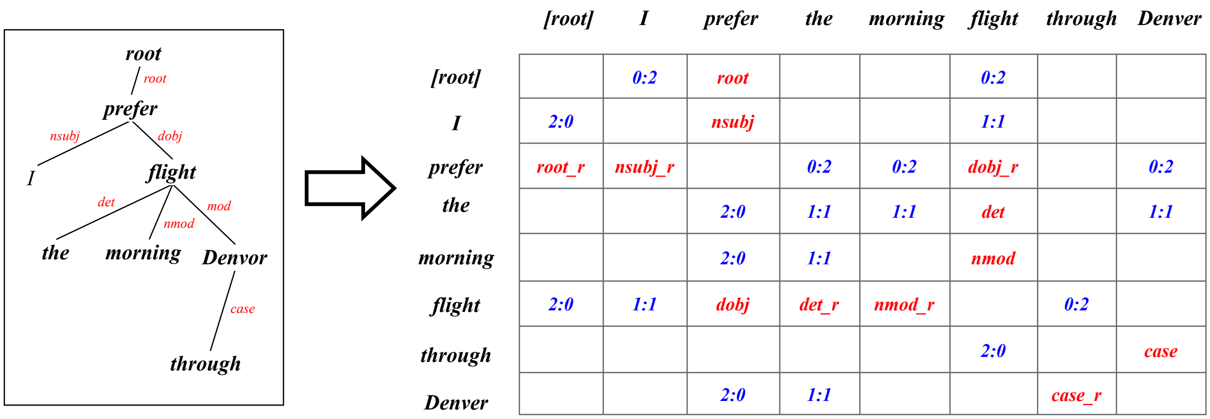
$$S^r_{ij} = r_{ij} V_r \ , \tag{1}$$

where $r_{ij} \in \mathbb{R}^{d_r}$ denote the relation vector of $word_i$ to $word_j$, $V_r \in \mathbb{R}^{d_r\times 1}$ is a learnable vector in current attention head.

Noted that, in the current mechanism, the score of relation-attention and self-attention are independent of each other. This suggests that these scores only depend on the tree, limiting the model's fitting ability to some degree. Thus, we expect this score to be semantically correlated. Give a simple example, two phrases: ***a cute girl*** and ***a little girl***. In a sentiment analysis task, intuitively, even they have the same type of dependency relations, the score of ***girl*** attend to ***little*** should be lower than ***girl*** attend to ***cute*** because ***cute*** expresses a stronger sentiment. This suggests that these scores should be associated with dependents. With respect to these cases, we design a gating mechanism to scale relation-attention scores. That is, a gated score calculated by relation and query word determines the proportion of relation-attention score that will be added to the attention score. We pool the relation embeddings and hidden states to gated scores and use the sigmoid function to scale them to $g \in [0, 1]$. We reformulate the attention score as follows:

$$S_{ij} = S^e_{ij} + S^r_{ij} \ , \tag{2}$$

$$S^r_{ij} = g_{ij} \odot (r_{ij} V_r) \ , \tag{3}$$



	[root]	I	prefer	the	morning	flight	through	Denver
[root]		0:2	root			0:2		
I	2:0		nsubj			1:1		
prefer	root_r	nsubj_r		0:2	0:2	dobj_r		0:2
the			2:0	1:1	1:1	det		1:1
morning			2:0	1:1		nmod		
flight	2:0	1:1	dobj	det_r	nmod_r		0:2	
through						2:0		case
Denver			2:0	1:1			case_r	

Fig. 2. An example of relation matrix' constitution. Each element in the matrix represents the relation between two words. For the adjacent nodes, we assign the relations to the arc labels in the dependency tree(red color). For the non-adjacent nodes and the sum of their distances is less than a certain threshold(2 in the example), we concatenate their distances to the nearest common ancestor to represent the relations(blue color).

$$g_{ij} = sigmoid((h_i W_{g,e} + r_{ij} W_{g,r}) V_g) \ , \qquad (4)$$

where $W_{g,r} \in \mathbb{R}^{d_r \times d_r}$, $W_{g,e} \in \mathbb{R}^{d_e \times d_r}$, $V_g \in \mathbb{R}^{d_r \times 1}$ are trainable matrix and vectors which is independent in each attention heads. h_i denote the hidden states of $word_i$, the hidden states here are the sum of word embeddings, position embeddings and level embeddings in the first layer: $h_k^0 = e_k + p_k + l_k$, are the output of last layer in the other layers. And S_{ij}^e denotes the Transformer attention score of the corresponding word pair.

Nevertheless, if we only consider adjacent nodes in the tree structure, the relation matrix is very sparse since every dependent only has one head, and we cannot take advantage of the information contained in these tree structures. In order to enrich the relation type, we identify non-adjacent word pairs whose distance is less than a certain threshold and view them as dependency relations. In the implementation, we use a relative position encoding of shortest paths[14] to represent the relation between non-adjacent nodes. The specific construction of the relation matrix R is as shown in Figure 2((r_{ij} corresponds to the element of the i-th row and j-th column of R).

3. EXPERIMENTS

3.1. Experiment Setup

Dataset We evaluate the proposed model on three typical sentence representations tasks: text classification, text semantic matching and paraphrase detection. Text classification: Stanford Sentiment Treebank[9] for sentiment analysis in binary(SST-2). Text semantic matching: The Sentences Involving Compositional Knowledge dataset(SICK)[19] contains two tasks: relatedness task (SICK-R) and the entailment task (SICK-E). Paraphrase detection: Microsoft Research Paraphrase Corpus(MRPC)[20].

Table 1. Result of evaluation on four datasets.

Models	SICK-R	SICK-E	SST-2	MRPC
	(MSE)	(Acc.)(%)	(Acc.)(%)	(Acc.)(%)
LSTM[12]	.2831	76.80	84.90	71.70
BiLSTM[12]	.2736	82.11	87.50	72.70
RNTN[10]	-	59.42	85.40	66.91
DT-RNN[10]	.3848	63.38	86.60	67.51
Tree-LSTM$_{DT}$[12]	.2734	82.00	85.70	72.07
Tree-LSTM$_{CT}$[12]	.2532	83.11	88.00	70.07
DC-TreeLSTM[15]	-	82.30	87.80	-
BiTree-LSTM[16]	.2736	-	90.30	-
TagHyperTreeLSTM[17]	-	83.90	**91.20**	-
USE[18]	-	81.15	85.38	**74.96**
Tree-Transformer$_{DT}$[13]	.2774	82.95	83.12	70.34
Tree-Transformer$_{CT}$[13]	.3012	82.72	86.66	71.73
Ours	**.2428**	**85.10**	88.70	73.58

Baselines We compare our model with several state-of-the-art models, including four basic architectures: LSTM and Bi-LSTM[12] , and eight tree-based models: DT-RNN[10], Tree-LSTM[12], Bi-treeLSTM[16] , DC-treeLSTM[15], TagHyperTreeLSTM[21], and two Transformer-based model: USE[18] and Tree-Transformer[13]. Bi-treeLSTM is a encoder added bidirectional flow to tree-LSTM. DC-treeLSTM and TagHyperTreeLSTM took advantage of hypernetworks to generate dynamic parameters for different inputs. USE is a pretrained sentence encoder based on Transformer. For Tree-

Table 2. Training and testing time of Dependency-Transformer and Tree-LSTM(Pytorch impletation) in one epoch on SICK-E.

Model	Testing	Training
Tree-LSTM	8s	282s
Ours	<1s	57s

LSTM and Tree-Transfomer, they include two version which based on dependency tree or constituency tree respectively, and we distinguish them with subscripts CT and DT.

Setting For all tasks, we use the Stanford dependency parser[22] to parse every sentence in datasets. Word embeddings are initialized with the 300-dimensional GloVe word vectors[23], and their weights will update during training. Our model consists of three layers, each of which uses a self-attention and relation-attention mechanism with six heads. We set the dimension of the relation vector to 30 and set the size of hidden states and Feed-Forward layers to 300. We take the output of $[root]$ token as the representation of this sentence. Our models are trained using AdaGrad[24] with a learning rate of 1e-3 and a batch size of 32. We report mean scores over five runs and trained our model on an Quadro RTX 5000 GPU, and used PyTorch 1.4.0 for the implementation.

For the text semantic matching task and paraphrase detection task, we use the same relatedness head module as tree-LSTM[12], which computes the final output of sentence pair by both the distance and angle of the two sentence's representation. For the classification task, including SICK-E, SST and MRPC, we use cross-entropy loss and report accuracy. For SICK-R, we use a KL-divergence loss to measure the distance between the predicted and target distribution report Pearson correlation between prediction and gold label. The results are shown in Table 1.

3.2. Result

For the SICK-E and SICK-R, our model is superior to all the baseline models and outperforms the top-performing models TreeLSTM and TagHyperTreeLSTM by an obvious margin: 0.0103 and 1.2%. For the SST task, our model achieves comparable accuracy with TagHyper-TreeLSTM and Bi-Tree-LSTM and is better than the other tree-based models. Our model also obviously outperforms USE and Tree-Transformer, which are also based on Transformer. In the SST task, the model receives supervision at all the nodes in the tree, and the number of nodes in the dependency tree is less than the constituency tree[12], which may limit the performance of our model on this dataset. For the MRPC task, our model is second only to USE and outperforms all the other tree-based models.

We test the time cost of our model and tree-LSTM on SICK-E, shown in Tab 2. As can be seen from the table, our training and reasoning time is significantly less than Tree-LSTM, which indicates that parallelism improves the efficiency of our model significantly.

Table 3. Performance of Dependency-Transformer with different implementation.

Models	**SICK-R**	**SICK-E**	**SST-2**	**MRPC**
	(MSE)	(Acc.)(%)	(Acc.)(%)	(Acc.)(%)
Transformer	0.2833	83.34	87.19	72.41
DT$_{lr}$	0.2545	84.76	88.21	73.01
DT$_{rg}$	0.2526	84.93	88.53	73.11
DT$_{full}$	0.2428	85.13	88.77	73.58

3.3. Ablation Study

We conduct an ablation study to validate the proposed mechanism. As shown in Tab 3, we compare the results of four models with different implementations. In the table, $Transfomer$ refers to the Transformer encoder with the same setting as our model. DT_{lr} refers to Dependency-Transformer without level embedding, and gating mechanism, DT$_{rg}$ refers to Dependency-Transformer without the level embedding, DT_{full} refers the full implementation of the Dependency-Transformer by adding gating mechanism to relation-attention. Clearly, the relation-attention mechanism enhances the original Transformer's performance significantly. It proves that the label of relation and the spatial position all contribute to the tasks and the effectiveness and rationality of the relation-attention mechanism and gating mechanism.

4. CONCLUSION

We propose Dependency-Transformer, using a relation-attention mechanism that integrates dependency tree information into the self-attention mechanism. Additionally, we introduce a gating mechanism to link syntactic relation and semantic information. The model retains the parallel computing ability of the Transformer while enhanced by syntactic information. We show apparent advantages of the model in efficiency and performance on various sentence representation tasks compared to baselines.

Ackonwlegdement

This work is supported by National Key R&D Program of China (No.2017YFB1402405-5), and the Fundamental Research Funds for the Central Universities (No.2020CDCGJSJ0042). The authors thank all anonymous reviewers for their constructive comments.

5. REFERENCES

[1] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, “Supervised learning of universal sentence representations from natural language inference data,” *arXiv preprint arXiv:1705.02364*, 2017.

[2] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, “A structured self-attentive sentence embedding,” *arXiv preprint arXiv:1703.03130*, 2017.

[3] Y. Liu, C. Sun, L. Lin, and X. Wang, “Learning natural language inference using bidirectional lstm model and inner-attention,” *arXiv preprint arXiv:1605.09090*, 2016.

[4] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences,” *arXiv preprint arXiv:1404.2188*, 2014.

[5] Y. Kim, “Convolutionalneuralnetworksforsentence classification.”

[6] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.

[7] H. Zhao, Z. Lu, and P. Poupart, “Self-adaptive hierarchical sentence model,” *arXiv preprint arXiv:1504.05070*, 2015.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *arXiv preprint arXiv:1706.03762*, 2017.

[9] R. Socher, C. C.-Y. Lin, A. Y. Ng, and C. D. Manning, “Parsing natural scenes and natural language with recursive neural networks,” in *ICML*, 2011.

[10] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[11] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng, “Grounded compositional semantics for finding and describing images with sentences,” *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 207–218, 2014.

[12] K. S. Tai, R. Socher, and C. D. Manning, “Improved semantic representations from tree-structured long short-term memory networks,” *arXiv preprint arXiv:1503.00075*, 2015.

[13] M. Ahmed, M. R. Samee, and R. E. Mercer, “You only need attention to traverse trees,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 316–322, 2019.

[14] Y. Yang, Y. Tong, S. Ma, and Z.-H. Deng, “A position encoding convolutional neural network based on dependency tree for relation classification,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 65–74, 2016.

[15] P. Liu, X. Qiu, and X. Huang, “Dynamic compositional neural networks over tree structure,” *arXiv preprint arXiv:1705.04153*, 2017.

[16] Z. Teng and Y. Zhang, “Head-lexicalized bidirectional tree lstms,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 163–177, 2017.

[17] C. Xu, H. Wang, S. Wu, and Z. Lin, “Treelstm with tag-aware hypernetwork for sentence representation,” *Neurocomputing*, vol. 434, pp. 11–20, 2021.

[18] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Céspedes, S. Yuan, C. Tar, *et al.*, “Universal sentence encoder,” *arXiv preprint arXiv:1803.11175*, 2018.

[19] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, “Semeval-2014 task 1: Evaluation of compositional distributional,” *SemEval-2014*, 2014.

[20] W. Dolan, C. Quirk, C. Brockett, and B. Dolan, “Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources,” 2004.

[21] D. Ha, A. Dai, and Q. V. Le, “Hypernetworks,” *arXiv preprint arXiv:1609.09106*, 2016.

[22] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit,” in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60, 2014.

[23] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[24] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization.,” *Journal of machine learning research*, vol. 12, no. 7, 2011.