Modelling Sentence Pairs with Tree-structured Attentive Encoder

Yao Zhou, Cong Liu and **Yan Pan**
School of Data and Computer Science, Sun Yat-sen University
{zhouyao@mail2, liucong3@mail, panyan5@mail}.sysu.edu.cn

Abstract

We describe an attentive encoder that combines tree-structured recursive neural networks and sequential recurrent neural networks for modelling sentence pairs. Since existing attentive models exert the attention on the sequential structure, we propose a way of incorporating attention into the tree topology. Specially, our attentive encoder uses the representation of one sentence in the pair learned by a sequential recurrent nets to recursively guide the representation learning of the other sentence. We evaluate the proposed attentive encoder on three tasks: semantic similarity, paraphrase identification and true-false question selection. Experimental results show that our encoder outperforms all baselines and achieves state-of-the-art results on two tasks.

1 Introduction

The task of modelling a sentence pair is to score two pieces of sentences in terms of their semantic relevance. In particular, this task includes semantic relatedness (Marelli et al., 2014), paraphrase identification (He et al., 2015), answer selection (Yin et al., 2015) and textual entailment (Bowman et al., 2015; Rocktäschel et al., 2016) etc.

We consider that the approach of modelling a sentence pair based on neural networks can be divided into two steps. First, a sentence encoder transforms each sentence into a vector representation. Second, a classifier receives two sentence representations as features to make the classification. The sentence encoder can be regarded as a semantic compositional function which maps a sequence of word vectors to a sentence vec-

tor. This compositional function takes a range of different forms, including (but not limited to) sequential recurrent neural networks (Seq-RNNs) (Mikolov, 2012), tree-structured recursive neural networks (Tree-RNNs) (Socher et al., 2014; Tai et al., 2015) and convolutional neural networks (CNNs) (Kalchbrenner et al., 2014; Kim, 2014).

We introduce an approach that combines recursive neural networks and recurrent neural networks with the attention mechanism, which has been widely used in sequence to sequence learning (seq2seq) tasks ranging from machine translation (Bahdanau et al., 2015; Luong et al., 2015), text summarization (Rush et al., 2015) to natural language conversation (Shang et al., 2015) and other NLP tasks such as question answering (Sukhbaatar et al., 2015; Hermann et al., 2015), classification (Rocktäschel et al., 2016; Shimaoka et al., 2016). In the seq2seq learning, the attention mechanism is used to learn the alignments between source words and target words in the decoding phase. More generally, we consider that the motivation of attention mechanism is to allow the model to attend over a set of elements with the intention of attaching different emphases to each element. We argue that the attention mechanism used in a tree-structured model is different from a sequential model. Inspired from Rocktäschel et al. (2016) and Hermann et al. (2015), we use the attention mechanism to select semantically more relevant child by the representation of one sentence, which is learned by a Seq-RNNs, when constructing the head representation of the other sentence on a dependency tree. Since our model adopts the attention in the sentence encoding phase, we refer to it as an attentive encoder. In this work, we implement this attentive encoder with two architectures: tree-structured LSTM and GRU.

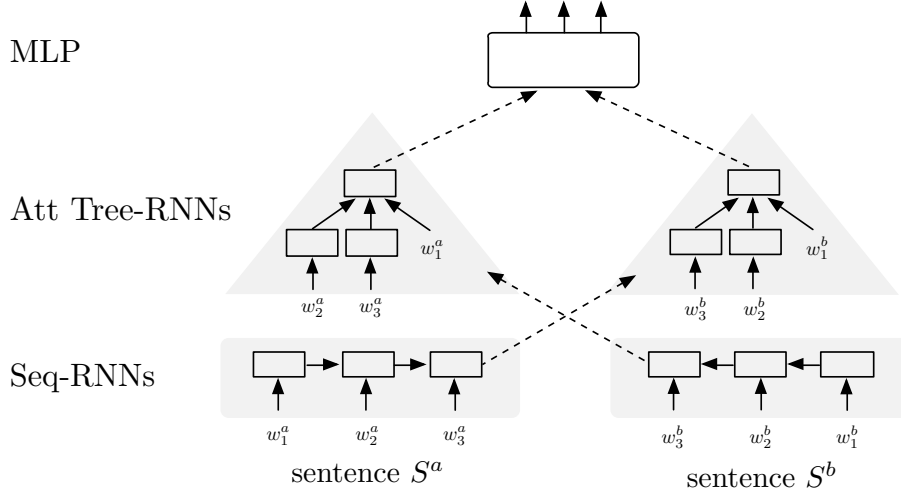We evaluate the proposed encoder on three sen-

Figure 1: An overview of our tree-structured attentive encoder.

tence pair modelling tasks: semantic similarity on the SICK dataset, paraphrase identification on the MSRP dataset and true-false question selection on the AI2-8grade science questions dataset. Experimental results demonstrate that our attentive encoder is able to outperform all non-attentional counterparts and achieves the state-of-the-art performance on the SICK dataset and AI2-8grade dataset.

2 Models

We begin with a high-level discussion of our tree-structured attentive encoder. As shown in Figure 1, given a sentence pair (S^a, S^b), our goal is to score this sentence pair. Our tree-structured attentive encoder has two components. In the first component, a pair of sentences is fed to the Seq-RNNs, which encodes each sentence and results in a pair of sentence representations. The second component (middle) uses the Att(entive) Tree-RNNs to encode the sentences again, aimed with the representation of the other sentence generated by the first component. The difference with previous models is that our attentive encoder consider not only the sentence itself but also its corresponding sentence. Finally, the two sentence vectors produced by the second component are fed to the multilayer perceptron network (MLP) to score the sentence pair. In the following, we will detailly introduce these components.

2.1 Seq-RNNs

We first describe the RNN composer, which is the basic unit of Seq-RNNs. Given an input sequence

of arbitrary length, an RNN composer iteratively computes a hidden state h_t using the input vector x_t and its previous hidden state h_{t-1}. In this paper, the input vector x_t is a word vector of the t-th word in a sentence. The hidden state h_t can be interpreted as a distributed representation of the sequence of tokens observed up to time t. Commonly, the RNN transition function is the following:

$$h_t = \tanh(Wx_t + Uh_{t-1} + b) \qquad (1)$$

We refer to the model that recursively apply the RNN composer to a sequence as the Seq-RNNs. Unfortunately, the standard Seq-RNNs suffers the problem that the gradients of the memory of earlier part of the sequence vanishes in long sequences (Hochreiter, 1998). Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Chung et al., 2014) are two powerful and popular architectures that address this problem by introducing gates and memory. In this paper, we only show the illustrations of LSTM (Figure 2(a)) and GRU (Figure 2(d)). We employ the same transition equations as (Tai et al., 2015) and (Chung et al., 2014). When we replace the standard RNN composer with LSTM or GRU, the Seq-RNNs becomes Seq-LSTMs or Seq-GRUs.

2.2 Standard Tree-RNNs

Comparison to standard RNN composer which computes its hidden state from the input at the current time step and the hidden state of previous time step, the Tree-RNN composer computes its hidden state from an input and the hidden states
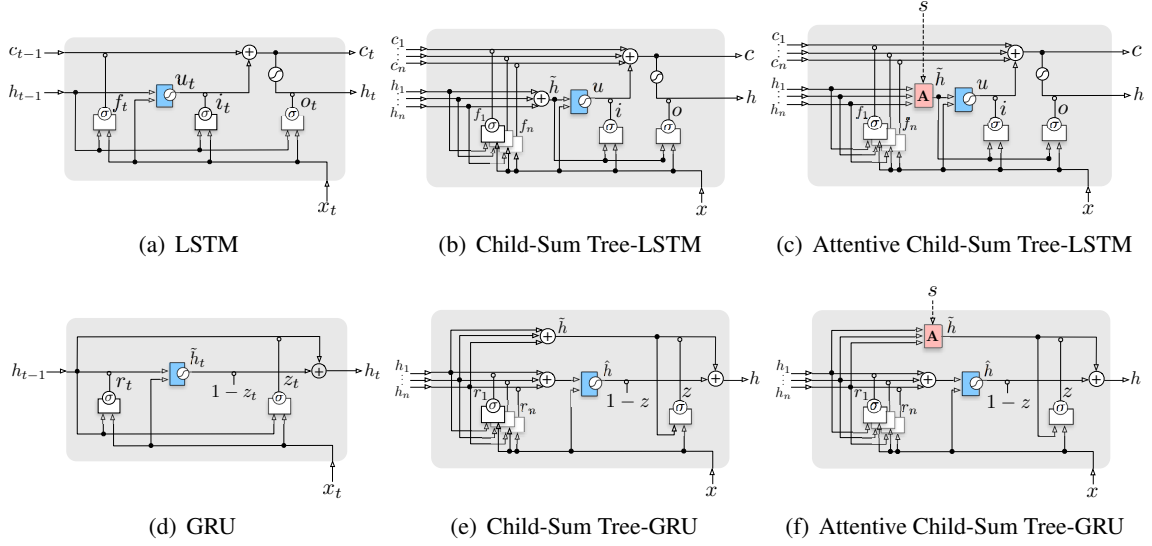
Figure 2: Illustrations of different composer architectures.

of arbitrarily many child units. We now describe the *Child-Sum Tree-LSTM* and *Child-Sum Tree-GRU* composer which are formed by applying the *Child-Sum* algorithm to LSTM and GRU respectively.

Child-Sum Tree-LSTM. In this paper, the implementation of Child-Sum Tree-LSTM is same as (Tai et al., 2015). We consider that a Child-Sum Tree-LSTM composer contains two parts: the external part and internal part. The external part consists of the inputs and outputs, and the internal part is the controllers and memory of the composer. As shown in Figure 2(b), the inputs of the composer are: a input vector x, multiple hidden states $h_1, h_2, \ldots, h_n$ and multiple memory cells $c_1, c_2, \ldots, c_n$, where n is the number of child units. The outputs consist of a memory cell c and a hidden state h which can be interpreted as the representation of a phrase. The internal part aims at controlling the flow of information by an input gate i, an output gate o and multiple forget gates $f_1, f_2, \ldots, f_n$. The gating mechanisms used in the Child-Sum Tree-LSTM are similar to sequential LSTM. Intuitively, the sum of children's hidden states $\tilde{h}$ is the previous hidden state, the forget gate f_k controls the degree of memory forgotten of the child k, the input gate i controls how much the internal input u is updated and the output gate controls the exposure of internal memory c. We define the transition equations as following:

$$
\begin{aligned}
\tilde{h} &= \sum_{1 \le k \le n} h_k, \\
i &= \sigma(W^{(i)} x + U^{(i)} \tilde{h} + b^{(i)}), \\
o &= \sigma(W^{(o)} x + U^{(o)} \tilde{h} + b^{(o)}), \\
u &= \tanh(W^{(u)} x + U^{(u)} \tilde{h} + b^{(u)}), \quad (2) \\
f_k &= \sigma(W^{(f)} x + U^{(f)} h_k + b^{(f)}), \\
c &= i \odot u + \sum_{1 \le k \le n} f_k \odot c_k, \\
h &= o \odot \tanh(c),
\end{aligned}
$$

Child-Sum Tree-GRU. We implement the Child-Sum Tree-GRU by the way of Child-Sum Tree-LSTM extending to the standard LSTM. Since we only introduce the Child-Sum algorithm applied to the LSTM and GRU, in the following, we omit "Child-Sum" prefix of Child-Sum Tree-LSTM and Child-Sum Tree-GRU for simplicity. Comparison to the Tree-LSTM, the Tree-GRU removes the memory cell c and introduces an update gate z and multiple reset gates $r_1, r_2, \ldots, r_n$ that allow the composer to reset the hidden state of child units. The candidate hidden state $\hat{h}$ is computed similarly to the standard RNN (Equation 1) and the update gate z is used to control how much the previous hidden state $\tilde{h}$ and the candidate $\hat{h}$ should be passed. The transition equations of Tree-GRU are the following:

$$
\begin{aligned}
\tilde{h} &= \sum_{1 \le k \le n} h_k, \\
r_k &= \sigma(W^{(r)} x + U^{(r)} h_k + b^{(r)}), \\
\hat{h} &= \tanh(W^{(h)} x + U^{(h)} \sum_{k=1}^{n} r_k \odot h_k), \\
z &= \sigma(W^{(z)} x + U^{(z)} \tilde{h} + b^{(z)}) \\
h &= z \odot \tilde{h} + (1 - z) \odot \hat{h},
\end{aligned}
$$

(3)

where σ denotes the logisitic sigmoid function and $\odot$ denotes element-wise multiplication.

We can easily apply the Child-Sum Tree-RNN to the dependency tree which has branch factors of arbitrary number and whose children are order-insensitive. We refer to the model adopting the Tree-LSTM and Tree-GRU composer to the dependency tree as the *Dependency Tree-LSTMs* and *Dependency Tree-GRUs*. For simplicity, we also omit the prefix "Dependency" in the following sections.

2.3 Attentive Tree-RNNs

We now describe the details how we extend the standard Tree-RNN. The idea we incorporate the attention into the standard Tree-RNN comes from: (1) there will be latent semantic relevance between two sentences in the sentence pair modelling tasks; (2) the effect of semantic relevance could be implemented in the process of constructing the sentence representation by Tree-RNN where each child should be assigned with a different weight; and (3) the attention mechanism is well suited for learning weights for a set of vectors when using a guided vector to attend over them.

Soft Attention Layer In this work, the attention mechanism is implemented by a *soft attention layer* A. Given a collection of hidden states $h_1, h_2, \ldots, h_n$ and an external vector s, the soft attention layer produce a weight α_k for each hidden state as well as a weighted vector g via the Equations 4:

$$
\begin{aligned}
m_k &= \tanh(W^{(m)} h_k + U^{(m)} s), \\
\alpha_k &= \frac{\exp(w^\mathsf{T} m_k)}{\sum_{j=1}^n \exp(w^\mathsf{T} m_j)}, \quad (4) \\
g &= \sum_{1 \le k \le n} \alpha_k h_k,
\end{aligned}
$$

Attentive Tree-LSTM and -GRU As illustrated in Figure 2(c) and Figure 2(f), when the attention layer is embedded to the standard Tree-LSTM and Tree-GRU composer, these composers become the Attentive Tree-LSTM and Attentive Tree-GRU. The attention layer (notated with A) receives the children's hidden states and an external vector s, producing a weighted representation g (Equation 4). In our implementation, the external vector s is a vector representation of sentence learned by a Seq-RNNs. Specially, if the tree composer is Attentive Tree-LSTM, then the Seq-RNNs is Seq-LSTMs, else if Attentive Tree-GRU then Seq-GRUs. Instead of taking the sum of children's hid-

den states as the previous hidden state $\tilde{h}$ in the standard Tree-RNN, we compute a new hidden state by a transformation $\tilde{h} = \tanh(W^{(a)} g + b^{(a)})$. Similar to the standard Tree-RNN, the attentive composers also can be easily applied to the dependency tree. We refer to the model which applies the Attentive Tree-RNN composer to the dependency tree as Attentive (Dependency) Tree-RNNs.

2.4 MLP

The multilayer perceptron network (MLP) receives a pair of vectors produced by the sentence encoder to compute a probability distribution. Given two sentence representations h_L and h_R, we compute their componentwise product $h_L \odot h_R$ and their absolute difference $|h_L - h_R|$. These features are also used by (Tai et al., 2015). We then compress these features into a low dimensional vector h_s, which is used to compute the probability distribution $\hat{p}_\theta$. The equations are the following:

$$
\begin{aligned}
h_\times &= h_L \odot h_R, \quad h_+ = |h_L - h_R|, \\
h_s &= \sigma(W^{(\times)} h_\times + W^{(+)} h_+ + b^{(h)}), \\
\hat{p}_\theta &= \begin{cases} \text{softmax}(W^{(p)} h_s + b^{(p)}) & \text{if classes} > 2, \\ \text{sigmoid}(W^{(p)} h_s + b^{(p)}) & \text{if classes} = 2, \end{cases}
\end{aligned}
$$
$$(5)$$

3 Experiments and Results

Our baselines In order to make a meaningful comparison between the sequential models, tree-structured models and attentive models, we present four baselines. They are: (i) **Seq-LSTMs**, learning two sentence representations by the sequential LSTMs; (ii) **Seq-GRUs**, like Seq-LSTMs but using GRU composer; (iii) **Tree-LSTMs**, learning two sentence representations by the Dependency Tree-LSTMs; and (iv) **Tree-GRUs**, like Tree-LSTMs but using Child-Sum Tree-GRU composer. The two sentence representations are fed to the MLP to produce the probability distribution of score of sentence pair.

3.1 Task 1: Semantic Similarity

First we conduct our semantic similarity experiment on the Sentences Involving Compositional Knowledge(SICK) dataset (Marelli et al., 2014). This task is to predict a similarity score of a pair of sentences, based on human generated scores. The SICK dataset consists of 9927 sentence pairs with

Config	Value	Config	Value
Word vectors	Glove (Pennington et al., 2014)	Dims of word vectors	300
OOV word vectors	uniform(-0.05, 0.05)	Dims of hidden state	150
Learning rate	0.05	Batch size	25
Regularization	L2 with $\lambda = 10^{-4}$	Dropout rate	0.5
Optim method	Adagrad (Duchi et al., 2011)	Num of epoch	10

Table 1: The training configs. We use the 300D Glove vectors as the initial word vectors. Out-of-vocabulary words are initialized with a uniform distribution. The model parameters are regularized with a per-minibatch L2 regularization strength of 10^{-4}. The dropout is used at the classifier with a dropout rate 0.5. All models are trained using Adagrad with a learning rate of 0.05. We train our models for 10 epochs, and use the model that has the best results on the dev set to evaluate on the test set.

the split of 4500 training pairs, 500 development pairs and 4927 testing pairs. Each sentence pair is annotated with a similarity score ranging from 1 to 5. A high score indicates that the sentence pair is highly related. All sentences are derived from existing image and video annotation dataset. The evaluation metrics are Pearson's r, Spearman's ρ and mean squared error (MSE).

Recall that the output of MLP (Section 2.4) is a probability vector $\hat{p}_\theta$. Our goal in this task is to predict a similarity score of two sentences. Let $r^{\mathsf{T}} = [1, \ldots, 5]$ be an integer vector, the similarity score $\hat{y}$ is computed by $\hat{y} = r^{\mathsf{T}}\hat{p}_\theta$. We use the same setup as (Tai et al., 2015) that computes a distribution p as a function of prediction score y given by:

$$p_i = \begin{cases} y - \lfloor y \rfloor, & i = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, & i = \lfloor y \rfloor \\ 0 & \text{otherwise} \end{cases}$$

The criterion of semantic similarity is the *DistKLDivCriterion* that measures the continuous distance between the prediction and the ground truth.

The results are summarized in Table 2. We first compare our results against the previous results. ECNU (Zhao et al., 2014), the best result of SemEval 2014 submissions, achieves a 0.8414 r score by a heavily feature-engineered approach. Kiros et al. (2015) presents an unsupervised approach to learn the universal sentence vectors without depending on a specific task. Their Combine–skip+COCO model improve the Pearson's r to 0.8655, but a weakness is that their sentence vectors are high-dimensional vectors (2400D). Training the skip-thoughts vectors needs a lot of time and space. He et al. (2015) show the effectiveness of convolutional nets with the similarity

measurement layer for modelling sentence similarity. Their ConvNet outperforms ECNU with +0.027 Pearson's r. We can observe that dependency Tree-LSTM, combine-skip+COCO and ConvNet almost achieve the same performance and our Attentive Tree-LSTMs outperforms these three methods around +0.005 points. Comparison to ECNU, our Attentive Tree-LSTMs gains an improvement of +0.032 and achieves the state-of-the-art performance. We find a phenomenon also appeared in (Tai et al., 2015) that tree-structured models can outperform sequential counterparts. Comparison to the non-attentional baselines (such as Tree-LSTMs), the attention mechanism (such as Attentive Tree-LSTMs) gives us a boost of around +0.007. All results highlight that our attentive Tree-RNNs are well suited for the semantic similarity task.

3.2 Task 2: Paraphrase Identification

The next task we evaluate is paraphrase identification on the Microsoft Research Paraphrase Corpus (MSRP) (Dolan et al., 2004). Given two sentences, this task is to predict whether or not they are paraphrases. The dataset is collected from news sources and contains 5801 pairs of sentences, with 4076 for training and the remaining 1725 for testing. We randomly select 10% of training set and use them as our dev set. This task is a binary classification task, therefore we report the accuracy and F1 score.

Note that the element of $\hat{p}_\theta$ indicates the predicted probability of the indexed class, therefore the class label is equal to $\text{argmax}(\hat{p}_\theta)$. For the binary classification task, the criterion is the *BCECriterion*.

Table 3 (**left**) presents our results on the MSRP dataset. The previous approaches are:

Method	r	ρ	MSE
ECNU (Zhao et al., 2014)	0.8414	-	-
Dependency Tree-LSTMs (Tai et al., 2015)	0.8676	0.8083	0.2532
combine-skip+COCO (Kiros et al., 2015)	0.8655	0.7995	0.2561
ConvNet (He et al., 2015)	0.8686	0.8047	0.2606
Seq-GRUs	0.8595	0.7974	0.2689
Seq-LSTMs	0.8528	0.7911	0.2831
(Dependency) Tree-GRUs	0.8672	0.8116	0.2573
(Dependency) Tree-LSTMs(ours)	0.8664	0.8068	0.2610
+Attention			
Attentive (Dependency) Tree-GRUs	0.8701	0.8085	0.2524
Attentive (Dependency) Tree-LSTMs	**0.8730**	**0.8117**	**0.2426**

Table 2: Test set results on the SICK dataset. The first group is previous results, and the second is ours.

Method	Acc(%)	F1(%)
Baseline (Mihalcea et al., 2006)	65.4	75.3
RAE (Socher et al., 2011)	76.8	83.6
combine-skip+feats (Kiros et al., 2015)	75.8	83.0
ABCNN-3 (Yin et al., 2015)	78.9	84.8
TF-KLD (Ji and Eisenstein, 2013)	**80.4**	**85.9**
Seq-GRUs	71.8	80.2
Seq-LSTMs	71.7	80.6
Tree-GRUs	73.6	81.8
Tree-LSTMs	73.5	82.1
+Attention		
Attentive Tree-GRUs	74.8	82.3
Attentive Tree-LSTMs	**75.8**	**83.7**

Method	Dev Acc(%)	Test Acc(%)
RNN (Baudis et al., 2016)	38.1	36.1
CNN (Baudis et al., 2016)	44.2	38.4
RNN-CNN (Baudis et al., 2016)	43.9	37.6
attn1511 (Baudis et al., 2016)	38.4	35.8
Ubu.RNN (Baudis et al., 2016)	49.4	44.1
Seq-GRUs	72.1	62.4
Seq-LSTMs	71.8	63.3
Tree-GRUs	75.2	70.6
Tree-LSTMs	74.6	69.1
+Attention		
Attentive Tree-GRUs	**76.4**	72.1
Attentive Tree-LSTMs	76.2	**72.5**

Table 3: The test results of paraphrase identification on the Microsoft Paraphrase Corpus and true-false selection on the AI2-8grade dataset.

(1) Baseline, cosine similarity with tf-idf weighting; (2) RAE, recursive autoencoder with dynamic pooling; (3) combine-skip+feats, skip-thought vectors with features; (4) ABCNN-3, attention-based convolutional nets; and (5) TF-KLD, matrix factorization with supervised reweighting. First, all our models are able to outperform the baseline. We only compare our models with the neural networks-based approaches, including RAE and ABCNN-3 for a fair comparison. We find that our models do not prove to be very competitive. After a careful analysis, we conclude that the reasons are (1) our models are pure neural networks-based, we don't add any features to identify paraphrases while the other methods have used additional features; (2) The MLP is not very suitable in this task. We attempt to use the cosine distance and euclidean distance to replace the MLP in our future work. Although our models have not yet matched the SOTA performance, we obtain an improvement of +2.3 accuracy by Attentive Tree-LSTMs when we incorporate the attention into the standard Tree-LSTM.

3.3 Task 3: True-False Question Selection

We last consider a challenging task: selecting true or false given a scientific question and its evidence. In this task, we use the AI2-8grade dataset built by (Baudis et al., 2016). This dataset is derived from the **AI2 Elementary School Science Questions** released by Allen Institute. Each sentence pair consists of a hypothesis sentence processed by substituting the wh-word in the question by answer and its evidence sentence extracted from a collection of CK12 textbooks. The number of sample pairs in the training, development, and test set are 12689, 2483 and 11359 respectively. This dataset contains 626 words not appearing in Glove vectors, most of which are named entities and scientific jargons.

We use the same criterion as the paraphrase identification since it is also a binary classification task. We reports the dev accuracy and test accuracy shown in Table 3 (**right**). Since this dataset is a fresh and uncompleted dataset, we only compare our models with Baudis et al. (2016) who have evaluated several models on it. Comparison to (Baudis et al., 2016), all of our models gain a

Dataset	Sentence 1	Sentence 2	GT	pred
	The black dog is playing with the brown dog on the sand	A black dog is playing with a brown dog on the sand	4.8	4.8
SICK	A brown dog and a black dog are playing in the sand	The black dog is playing with the brown dog on the sand	5.0	4.2
	A brown dog and a black dog are playing in the sand	A black dog is attacking a brown dog on the sand	3.5	3.4
	The study is being published today in the journal Science.	Their findings were published today in Science.	1	1
MSRP	The launch marks the start of a new golden age in Mars exploration.	The launch marks the start of a race to find life on another planet.	1	1
	Last year, Comcast signed 1.5 million new digital cable subscribers.	Comcast has about 21.3 million cable subscribers, many in the largest U.S. cities.	0	1
	Sunlight is the nutrient source for some fungi ?	The main difference between plants and fungi is how they obtain energy.	0	0
	Sunlight is the nutrient source for some fungi ?	Plants are autotrophs, meaning that they make their own "food" using the energy from sunlight.	0	0
AI2	Sunlight is the nutrient source for some fungi ?	Fungi are heterotrophs, which means that they obtain their "food" from outside of themselves.	0	0
	Dead organisms is the nutrient source for some fungi ?	Most fungi live in soil or dead matter, and in symbiotic relationships with plants, animals, or other fungi.	1	0
	Dead organisms is the nutrient source for some fungi ?	Relate the structure of fungi to how they obtain nutrients.	1	0
	Dead organisms is the nutrient source for some fungi ?	From dead plants to rotting fruit.	1	1

Table 4: Example predictions from the test set. GT = ground truth, pred = predicted score.

(a) SICK dataset
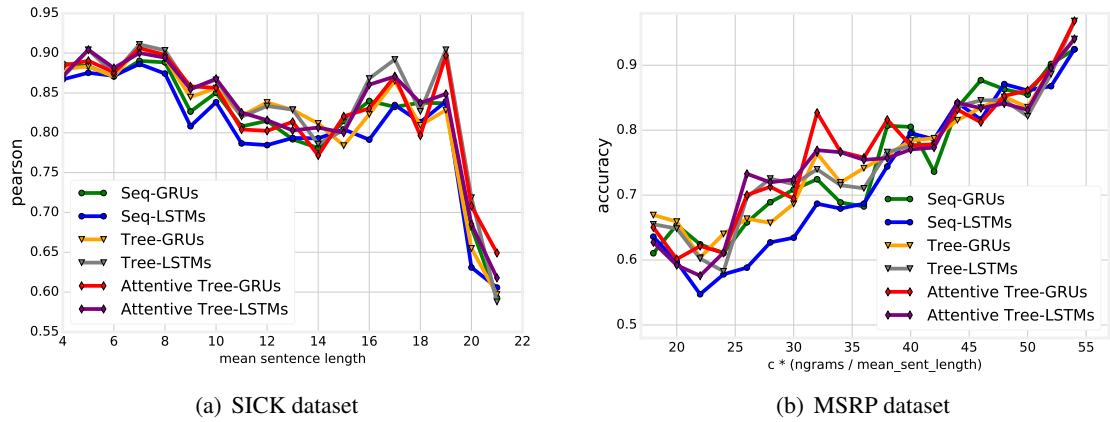

(b) MSRP dataset

Figure 3: Qualities of different models based on mean sentence length and ngrams

significant improvement. Specially, our best result achieved by the Attentive Tree-LSTMs is higher than the best of (Baudis et al., 2016) by +28 percents. We also observe that tree-structured models are more competitive than sequential counterparts. As we expected, the attentive models can outperform all non-attentional counterparts.

4 Quantitative Analysis

Example Analysis Table 4 presents example predictions that are produced by our Attentive Tree-LSTMs. The first group shows that our model is able to predict semantic similarity score nearly perfectly on the SICK dataset. We argue the reason is that the sentences of SICK dataset are image and video descriptions whose sentence structure is relatively simple and there are less uncommon words and named entities in the vocabulary. The second group gives us three examples on the MSRP test set. We find that our model can identify whether two fact statements are para-

phrases, but fails to recognize the numbers (in group 2, line 3). We presents the examples on AI2-8grade dataset in the last group. We can observe that our model is efficient to select the false questions, while our model is difficult to select the true answers, unless the evidence of question is very strong.

Effect of Sentence Length In order to analyse the effect of mean sentence length on the SICK dataset, we draw the Figure 3(a). We observe that the Pearson score become lower as sentence become longer. Compared with the Seq-RNNs, the Tree-RNNs obtain a little improvements. Specially, the Attentive Tree-GRUs proves to be more effective than Tree-GRUs when the mean sentence length reaches to 20.

Effect of N-grams In the paraphrase identification task, a hypothesis is that two sentences tend to be paraphrases when the value of their n-grams is high. As shown in Figure 3(b), xaxis is the
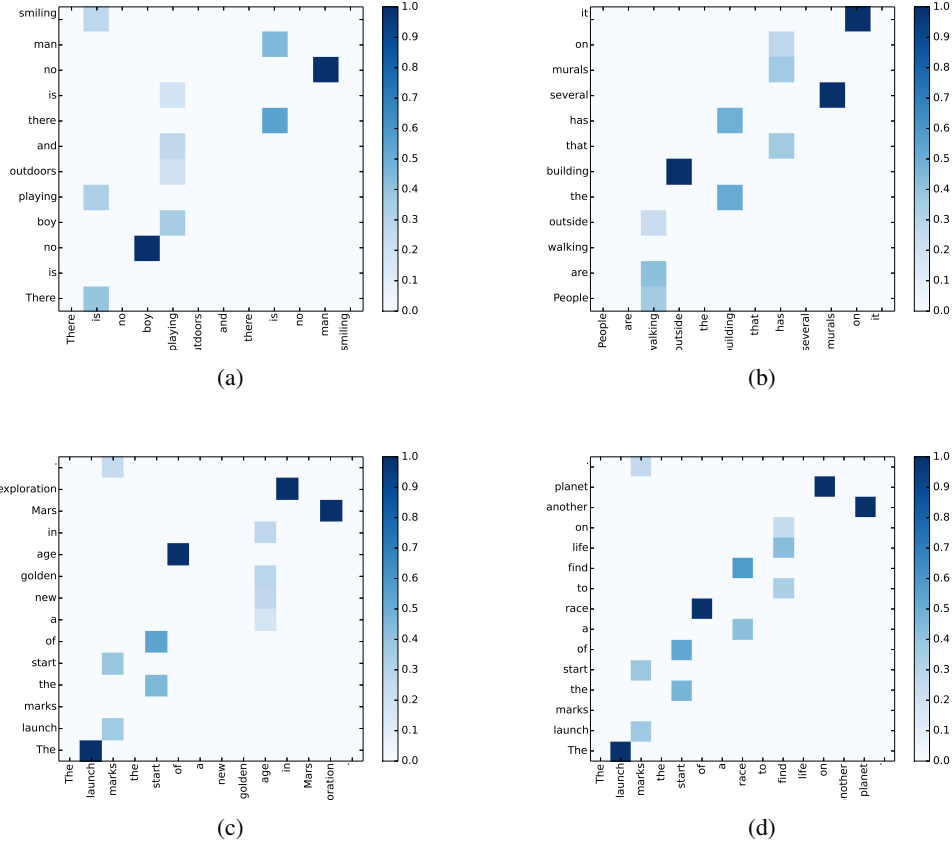
Figure 4: Heatmap of attention weights.

normalized n-grams whose value is computed by $c * \frac{(unigram+bigram+trigram)}{mean\_sent\_length}$, where c equals to 50, and yaxis is the accuracy. We observe that Attentive Tree-GRUs are more effective than Tree-GRUs when the value of n-grams is less than 40. It suggests that our attentive models are more general.

Attention Visualization It is instructive to analyze which child the attentive model is attending over when constructing the head representation. We visualize the heatmaps of attention weights shown in Figure 4. The xaxis word is modified by the yaxis words with a weight (greater than *zero*). For example in Figure 4(a), the xaxis word of the 5th column is "playing" whose children are "boy", "outdoors", "and" and "is". We can observe that the branch rooted with "boy" contributes more when constructing the representation of subtree whose root node is "playing".

5 Conclusion

In this paper, we introduced a way of incorporating attention into the Child-Sum Tree-LSTM and -GRU which can be applied to the dependency tree. We evaluate the proposed models on three sentence pair modelling tasks and achieve state-of-the-art performance on two of them. Experiment results show that our attentive models are effective for modelling sentence pairs. In the future, we will evaluate our models on the other sentence pair modelling tasks and extend this approach to seq2seq learning framework.

References

[Bahdanau et al.2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

[Baudis et al.2016] Petr Baudis, Silvestr Stanko, and Jan Sedivy. 2016. Joint learning of sentence embeddings for relevance and entailment. *arXiv preprint arXiv:1605.04655*.

[Bowman et al.2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015*

Conference on Empirical Methods in Natural Language Processing, pages 632–642.

[Chung et al.2014] Junyoung Chung, Çalar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. Technical Report Arxiv report 1412.3555, Université de Montréal. Presented at the Deep Learning workshop at NIPS2014.

[Dolan et al.2004] Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, page 350. Association for Computational Linguistics.

[Duchi et al.2011] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

[He et al.2015] Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586.

[Hermann et al.2015] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

[Hochreiter1998] Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

[Ji and Eisenstein2013] Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896.

[Kalchbrenner et al.2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 655–665.

[Kim2014] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

[Kiros et al.2015] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

[Luong et al.2015] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

[Marelli et al.2014] Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *SemEval-2014*.

[Mihalcea et al.2006] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780.

[Mikolov2012] Tomáš Mikolov. 2012. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April*.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, volume 14, pages 1532–43.

[Rocktäschel et al.2016] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *International Conference on Learning Representations*.

[Rush et al.2015] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

[Shang et al.2015] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1577–1586.

[Shimaoka et al.2016] Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, San Diego, California, USA, June. to appear.

[Socher et al.2011] Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y

Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

[Socher et al.2014] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218.

[Sukhbaatar et al.2015] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc.

[Tai et al.2015] 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 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1556–1566.

[Yin et al.2015] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*.

[Zhao et al.2014] Jiang Zhao, Tian Tian Zhu, and Man Lan. 2014. Ecnu: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. *Proceedings of the SemEval*, pages 271–277.