Reading and Thinking: Re-read LSTM Unit for Textual Entailment Recognition

Lei Sha, Baobao Chang, Zhifang Sui, Sujian Li
Key Laboratory of Computational Linguistics, Ministry of Education
School of Electronics Engineering and Computer Science, Peking University
Collaborative Innovation Center for Language Ability, Xuzhou 221009 China
{shalei, chbb, szf, lisujian}@pku.edu.cn

Abstract

Recognizing Textual Entailment (RTE) is a fundamentally important task in natural language processing that has many applications. The recently released Stanford Natural Language Inference (SNLI[1]) corpus has made it possible to develop and evaluate deep neural network methods for the RTE task. Previous neural network based methods usually try to encode the two sentences and send them together into multi-layer perceptron, or use LSTM-RNN to link two sentence together while using attention mechanic to enhance the model's ability. In this paper, we propose to use the intensive reading mechanic, which means to re-read the sentence (read the sentence again) according to the memory of the other sentence for a better understanding of the sentence pair. The re-read process can be applied alternatively between the two sentences. Experiments show that we achieve results better than current state-of-art equivalents.

1 Introduction

For the natural language, a common phenomenon is that there exist a lot of ways to express the same or similar meaning. To discover such different expressions, the Recognizing Textual Entailment (RTE) task is proposed to judge whether the meaning of one text (denoted as *hypothesis*) can be inferred (entailed) from the other one (*premise*) (Dagan et al., 2006). A simple example is shown in Table 1. For many natural language processing applications like question answering, information retrieval which need to deal with the diversity of natural language, recognizing textual entailments is a critical step.

Most previous neural network based methods are sentence encoding-based models. They applied a large variety of methods to encode the two sentences (premise and hypothesis), such as LSTM encoder, GRU encoder and tree-based CNN encoder. Then combine them as the feature of this sentence pair and send into a deep neural network for classification. However, in the sentence encoding process, the premise and the hypothesis cannot affect each other. It is well known that the encoding procedure is just automatically learning useful features. Without the impact between the two sentences, it is difficult for the encoder to extract the sentence-relationship-specific features. Other methods mainly make use of attention mechanism to capture the word-by word alignment information while training (Rocktäschel et al., 2015) or just integrate memory network into LSTM to make the model remember more information (Cheng et al., 2016). Among them, only the attention mechanism can make the two sentences contact with each other. However, the word-by-word attention does not represent a better understanding of the sentences.

When deciding the entailment relationship between a pair of sentences, what is really matters? Unlike paraphrasing and machine translation, entailment relationship does not force the two sentences have the same meaning. Instead, as long as the premise can cover the meaning of the hypothesis, the entailment stands. Therefore, if the premise entails the hypothesis, that doesn't really mean that the words in hypothesis can be totally entailed by the words in premise. For example, "these girls are having a great

[1]http://nlp.stanford.edu/projects/snli/

Relationship		Premise & Hypothesis
Entailment	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	The church is filled with song.
Neutral	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	The church has cracks in the ceiling.
Contradict	Premise:	This church choir sings to the masses as they sing joyous songs from the book at a church.
	Hypothesis:	A choir singing at a baseball game.

Table 1: Example of entailment / neutral / contradict eases.

time looking for seashells" can entail "the girls are outside". These entailment cases certainly cannot be solved by word-by-word alignment based methods since "outside" cannot be aligned to any of the words in the premise. Intuitively, when a human is judging the relationship of the two sentences, he/she would first read the premise, and then read the hypothesis while considering whether it can be entailed by the premise. Therefore, we intend to make the model more like human, namely, we require the model be capable of reading and thinking.

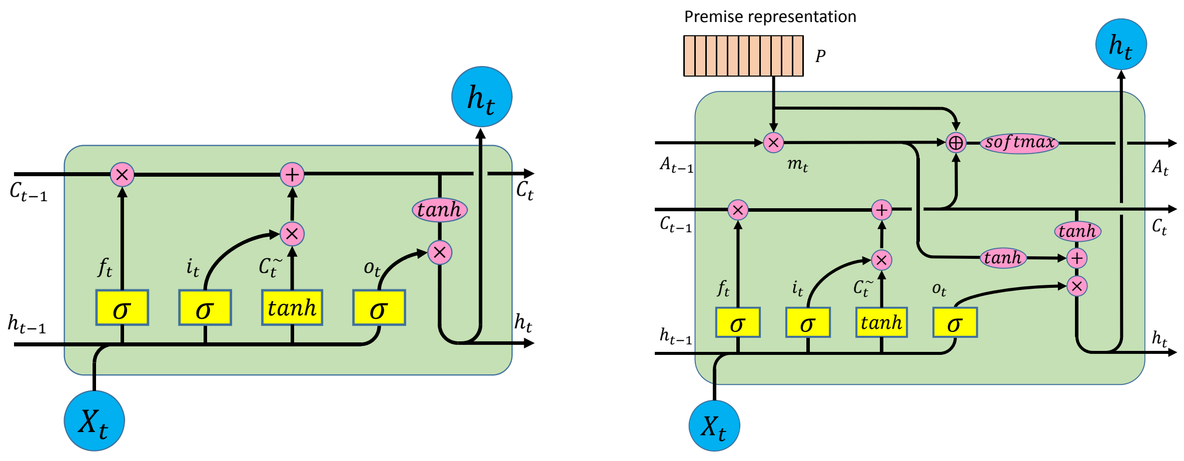
In this paper, we propose a new LSTM variant called re-read LSTM unit (rLSTM), which also take the attention vector of one sentence as an inner state while reading the other sentence. Therefore, this kind of unit is specially designed for dual sentence modeling. Then we use a standard bidirectional LSTM to read the premise, and use a bidirectional rLSTM to read the hypothesis. The output of the standard BiLSTM is taken as the general input of the bidirectional rLSTM. Experiments show that our method has outperformed the state-of-the-art approaches.

2 Related Work

Textual Entailment Recognizing (RTE) task has been widely studied by many previous work. Firstly, the methods use statistical classifiers which leverage a wide variety of features, including hand-engineered features derived from complex NLP pipelines and similarity between sentences (T and H) and sentence pairs ((T', H') and (T'', H''))(Malakasiotis and Androutsopoulos, 2007; Jijkoun and de Rijke, 2005; Wan et al., 2006; Zanzotto and Moschitti, 2006; Wang and Neumann, 2007; Dinu and Wang, 2009; Nielsen et al., 2009; Malakasiotis, 2011). This kind of methods are hard to generalize due to the complexity of feature engineering. Moreover, the hand-engineered features usually cannot represent implicit meanings of sentences.

Secondly, (Hickl, 2008; Sha et al., 2015; Shnarch et al., 2011b; Shnarch et al., 2011a; Beltagy et al., 2013; Rios et al., 2014) extract the structured information (discourse commitments or predicate-argument representations) in T-H pair and check if the information in T contains or can infer the information in H. Probabilistic methods are used for recognizing the entailment. However, these work are still based on hand-engineered features which is not easy to generalize.

Recently, neural network based methods start to show its effectiveness. Based on (Bowman et al., 2015), Rocktäschel et al. (2015) uses the attention-based technique to improve the performance of LSTM-based recurrent neural network. Then, Yin et al. (2015) applied attention mechanic to convolution neural network, Liu et al. (2016a) proposed coupled-LSTM, Vendrov et al. (2015) proposed ordered embedding, Mou et al. (2016) applied Tree-based CNN, Wang and Jiang (2015) proposed matching LSTM, Liu et al. (2016b) applied inner-attention, Cheng et al. (2016) proposed Long Short-Term Memory-Networks to improve the performance. To free the model from traditional parsing process, Bowman et al. (2016) combines parsing and interpretation within a single tree sequence hybrid model by integrating tree structured sentence interpretation into the linear sequential structure of a shift-reduce parser. Parikh et al. (2016) uses attention to decompose the problem into subproblems that can be solved separately, thus making it trivially parallelizable.



(a) LSTM Unit (b) rLSTM Unit

Figure 1: The inner architecture of the traditional LSTM unit and the re-read LSTM unit.

3 Model

3.1 Background

The LSTM architecture (Hochreiter and Schmidhuber, 1997) addresses this problem of learning long-term dependencies by introducing a memory cell that is able to preserve state over long periods of time. While numerous LSTM variants have been described, here we describe the most widely-used version.

Long short-term memory (LSTM) based recurrent neural networks (RNNs) have long been tried to apply to a wide range of NLP tasks, including RTE (Bowman et al., 2015). We define the LSTM unit at each time step t to be a collection of vectors in $\mathbb{R}^d$: an input gate i_t, a forget gate f_t, an output gate o_t, a memory cell c_t, candidate memory cell state $\widetilde{C}_t$ and a hidden state h_t. The entries of the gating vectors i_t, f_t and o_t are in $[0, 1]$. We refer to d as the memory dimension of the LSTM. The LSTM transition equations are listed in Eq 1.

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) & \widetilde{C}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) & c_t &= i_t \odot \widetilde{C}_t + f_t \odot c_{t-1} \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) & h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{1}$$

where x_t is the input at the current time step, σ denotes the logistic sigmoid function and $\odot$ denotes element-wise multiplication. Intuitively, the forget gate controls the extent to which the previous memory cell is forgotten, the input gate controls how much information is input to each unit, and the output gate controls the exposure of the internal memory state. The hidden state vector in an LSTM unit is therefore a gated, partial view of the state of the unit's internal memory cell. Since the value of the gating variables vary for each vector element, the model can learn to represent information over multiple time scales.

3.2 Re-read LSTM (rLSTM)

In the textual entailment recognition problem, we need to model the relationship of two sentences and judge whether the premise can entail the hypothesis. As for human beings, maybe the most nature way for the judging process is first read the premise, remember it, and then read the hypothesis while thinking whether the premise can entail the hypothesis. Intuitively, we can improve the performance of RTE by adding some human nature to deep neural network models. In this paper, we intend to make the LSTM unit capable of thinking the relation between premise and hypothesis while reading them. Our proposed LSTM architecture is shown in Figure 1b, we call it re-read LSTM (rLSTM). The rLSTM unit is specially designed for dual sentence modeling, it is applied only when dealing with the second sentence. Therefore, in Figure 1b, there is a general input to rLSTM: the premise representation. This representation is composed of each word's representation in the premise. The word's representation can

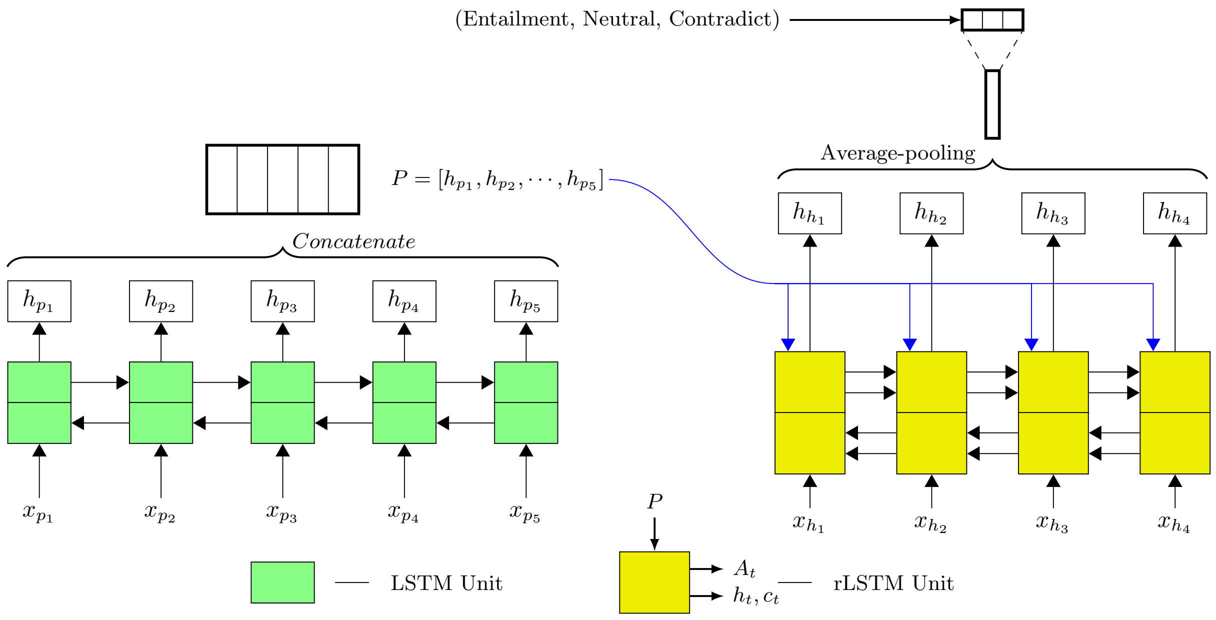

(Entailment, Neutral, Contradict)
Average-pooling
$P = [h_{p_1}, h_{p_2}, \cdots, h_{p_5}]$
Concatenate
h_{p_1} h_{p_2} h_{p_3} h_{p_4} h_{p_5}
x_{p_1} x_{p_2} x_{p_3} x_{p_4} x_{p_5}
h_{h_1} h_{h_2} h_{h_3} h_{h_4}
x_{h_1} x_{h_2} x_{h_3} x_{h_4}
LSTM Unit
P
A_t
h_t, c_t
rLSTM Unit

Figure 2: The architecture of our model.

be the word embedding or another plain LSTM's output. We add an input and an output to rLSTM unit which represents the attention over the words in the premise at time step $t-1$ (A_{t-1}) and $t(A_t)$. Given the attention at time $t-1$ and the premise representation P, we can get the current understanding of the premise, or the memory m_t:

$$m_t = A_{t-1}P \tag{2}$$

where $A_{t-1} \in \mathbb{R}^L$, $P \in \mathbb{R}^{d\times L}$, L represents the number of words in the premise.

The model needs to consider the entailment relationship. Therefore, the memory of the premise should also affect the hidden state of the hypothesis. With the information of premise, the hidden state of the hypothesis can learn more specific information for their relationship. So we add the memory of the premise and the memory cell to affect the hidden state vector as follows:

$$h_t = o_t \odot \Big(\tanh(c_t) + \tanh(m_t)\Big) \tag{3}$$

Intuitively, each time the model reads one more word in the hypothesis, it should be clearer about what information in the premise is more important respected to the hypothesis. So the A_{t-1} is the premise's attention in time step $t-1$, after read one word, it became A_t, the information represented by which is more focus on the relationship between the premise and the hypothesis. Therefore, the origin LSTM's memory cell can affect the attention in time step t:

$$\begin{aligned} \alpha &= W_pP + W_mm_t + W_cC_{t-1} \\ A_t &= softmax(w\alpha) \end{aligned} \tag{4}$$

where $W_p \in \mathbb{R}^{d\times d}$, $W_m \in \mathbb{R}^{d\times d}$, $W_c \in \mathbb{R}^{d\times d}$, $w \in \mathbb{R}^d$ are weight matrices or vectors.

3.3 Our Model

Our model is shown in Figure 2. We use the standard LSTM to deal with the premise. The output of the standard LSTM $h_{p_1}, h_{p_2}, \cdots, h_{p_n}$ are concatenated as a matrix P:

$$P = [h_{p_1}, h_{p_2}, \cdots, h_{p_L}] \tag{5}$$

where $P \in \mathbb{R}^{d\times n}$, L represents the number of words in the premise.

	Train	Dev	Test
Entailment	183416	3329	3368
Neutral	182764	3235	3219
Contradict	183187	3278	3237
Total	549367	9842	9824

Table 2: Distribution of Entailment Classes in SNLI

And then, we take P as the general input of re-read LSTM (rLSTM), which is used to deal with the hypothesis as is shown in Formula 6.

$$\begin{aligned} h_{\overrightarrow{h_i}} &= rLSTM(P, A_{t-1}, h_{h_{i-1}}, c_{t-1}) \\ h_{\overleftarrow{h_i}} &= rLSTM(P, A_{t+1}, h_{h_{i+1}}, c_{t+1}) \\ h_{h_i} &= [h_{\overrightarrow{h_i}}, h_{\overleftarrow{h_i}}] \end{aligned} \tag{6}$$

where the forward output $h_{\overrightarrow{h_i}}$ and the backward output $h_{\overleftarrow{h_i}}$ are all calculated by re-read LSTM unit. We concatenate the forward output $h_{\overrightarrow{h_i}}$ and the backward output $h_{\overleftarrow{h_i}}$ as the final output of the bidirectional rLSTM: h_{h_i}.

Then, we average the outputs of rLSTM as the final representation of the premise-hypothesis sentence pair:

$$S = \frac{1}{n} \sum_{i=1}^{n} h_{h_i} \tag{7}$$

where $S \in \mathbb{R}^d$ is the final representation. Then S is fed into a logistic classifier:

$$O = WS + b \tag{8}$$

where $W \in \mathbb{R}^{3 \times d}$, $b \in \mathbb{R}^3$ are the parameters, $O \in \mathbb{R}^3$ is the final output of this premise-hypothesis pair, each entry contains the score of an entailment class (entailment, neutral, contradiction).

3.4 Training

We define the ground-truth label vector y for each premise-hypothesis pair as a binary vector. If this premise-hypothesis pair belongs to class i, only the i-th dimension $y(i)$ is 1 and the other dimensions are set to 0. In our model, the RTE task is classification problem and we adopt cross entropy loss as the objective function. Given the parameters set $\theta = \{\theta_{LSTM}, \theta_{rLSTM}, W, b\}$, where θ_{LSTM} represents the LSTM neural network parameters, θ_{rLSTM} represents the rLSTM neural network parameters. the objective function for a premise-hypothesis pair can be written as,

$$J(\theta) = -\sum_{i} y(i) \log(O(i)) + \frac{\lambda}{2} \|\theta\|^2 \tag{9}$$

To compute the network parameter θ, we maximize the log likelihood $J(\theta)$ through stochastic gradient descent over shuffled mini-batches with the Adadelta (Zeiler, 2012) update rule.

4 Experiment

In this section, we present the evaluation of our model. We first perform quantitative evaluation, comparing our model with previous works. We then conduct some qualitative analyses to understand how our rLSTM model works in matching the premise and the hypothesis.

4.1 Datasets and Model Configuration

We conduct experiments on the Stanford Natural Language Inference corpus (SNLI) (Bowman et al., 2015). The original data set contains 570,152 sentence pairs, each labeled with one of the following relationships: entailment, contradiction, neutral and $-$, where $-$ indicates a lack of consensus from the human annotators. We discard the sentence pairs labeled with $-$ and keep the remaining ones for our experiments. Table 2 summarizes the statistics of the three entailment classes in SNLI.

We use 300 dimensional GloVe embeddings (Pennington et al., 2014) to represent words, which is trained on the Wikipedia+Gigaword dataset. The embeddings of unknown tokens are initialized by random vectors.

4.2 Methods for Comparison

Although we list all of the approaches designed for recognizing textural entailment task on the SNLI dataset in Table 3, we mainly want to compare our model with the word-by-word attention model by Rocktäschel et al. (2015), long short term memory network model by Cheng et al. (2016) and the decomposable attention model by Parikh et al. (2016) since they are either related to our work or achieved the state-of-the-art performance on the SNLI corpus.

We did the following ablation experiments:

- rLSTM: our final model.
- rLSTM - C info: in this model, in the rLSTM unit, the origin memory cell C_{t-1} cannot affect the next attention A_t. Then the Formula 4 is changed into Formula 10. We intend to see whether the origin memory cell's information can help the model to focus on more important information in the premise.

$$\begin{aligned} \alpha &= W_p P + W_m m_t \\ A_t &= softmax(w\alpha) \end{aligned} \tag{10}$$

- rLSTM - A info: in this model, in the rLSTM unit, the premise's memory m_t cannot affect the next hidden state h_t. Then the Formula 3 is changed into Formula 11, which is the origin formula of LSTM. We intend to see whether the current understanding of the premise can help the model to make better decision of the relation between the premise and the hypothesis.

$$h_t = o_t \odot \tanh(c_t) \tag{11}$$

- rLSTM - A&C info: in this model, in the rLSTM unit, the origin memory cell C_{t-1} cannot affect the next attention A_t and the premise's memory m_t cannot affect the next hidden state h_t. Then the LSTM part in the rLSTM is the same as the origin LSTM, and the attention part is updated each step only based on the model's understanding of the premise itself.

4.3 Quantitative Results

The experiment results are listed in Table 3. The experiment results are listed in Table 3. We can see that when we set d to 300 our final model (rLSTM) achieves an accuracy of 87.5% on the test data, which to the best of our knowledge is the highest on this data set. When we remove the effect of the origin memory cell C_{t-1} on the next attention A_t (rLSTM $-$ C info), we found that the performance has dropped 2.7 percent. This phenomenon shows that the memory of the hypothesis can indeed contribute to a better understanding of the premise. When we remove the effect of the premise's memory m_t on the next hidden state h_t, the performance dropped again. That means the premise's memory can indeed help understand the hypothesis.

When comparing our rLSTM model with Rocktäschel et al. (2015)'s LSTM word-by-word attention model under the same setting with $d = 100$, we can see that our performance on the test dataset is still higher than that of Rocktäschel et al. (2015)'s (**student t-test,** $p < 0.05$). One advantage of our model compared to Rocktäschel et al. (2015)'s model is that our attention is inside the LSTM unit, so that it can be affected by the inner memory of LSTM and can also affect the LSTM's inner state. This mechanic can provide our model more flexibility to achieve a better result.

Our result has also outperformed Cheng et al. (2016)'s 300d LSTMN model (**student t-test,** $p < 0.05$). LSTMN tends to use memory network to remember the sentence while making intra-attention (within a sentence) and inter-attention (between two sentences) complement each other to get good results.

Publication	Model	Params	Train	Test
Feature-based models				
Bowman et al. (2015)	Unlexicalized features	-	49.4	50.4
Bowman et al. (2015)	+ Unigram and bigram features	-	99.7	78.2
Sentence encoding-based models				
Bowman et al. (2015)	100D LSTM encoders	221k	84.8	77.6
Bowman et al. (2016)	300D LSTM encoders	3.0m	83.9	80.6
Vendrov et al. (2015)	1024D GRU encoders w/ unsupervised 'skip-thoughts' pre-training	15m	98.8	81.4
Mou et al. (2016)	300D Tree-based CNN encoders	3.5m	83.3	82.1
Bowman et al. (2016)	300D SPINN-PI encoders	3.7m	89.2	83.2
Liu et al. (2016b)	600D (300+300) BiLSTM encoders	2.0m	86.4	83.3
Liu et al. (2016b)	600D (300+300) BiLSTM encoders with intra-attention and symbolic preproc.	2.8m	85.9	85.0
Other neural network models				
Rocktäschel et al. (2015)	100D LSTMs w/ word-by-word attention	252k	85.3	83.5
Wang and Jiang (2015)	300D mLSTM word-by-word attention model	1.9m	92.0	86.1
Cheng et al. (2016)	300D LSTMN with deep attention fusion	1.7m	87.3	85.7
Cheng et al. (2016)	450D LSTMN with deep attention fusion	3.4m	88.5	86.3
Parikh et al. (2016)	200D decomposable attention model	382k	89.5	86.3
Parikh et al. (2016)	200D decomposable attention model with intra-sentence attention	582k	90.5	86.8
Re-read LSTM models				
This paper	300D rLSTM	2.0m	90.7	**87.5**
This paper	300D rLSTM - C info	1.9m	88.9	84.8
This paper	300D rLSTM - A info	2.0m	90.2	87.3
This paper	300D rLSTM - A&C info	1.9m	88.6	84.7

Table 3: Train/test accuracies on the SNLI dataset and number of parameters (excluding embeddings) for each approach.

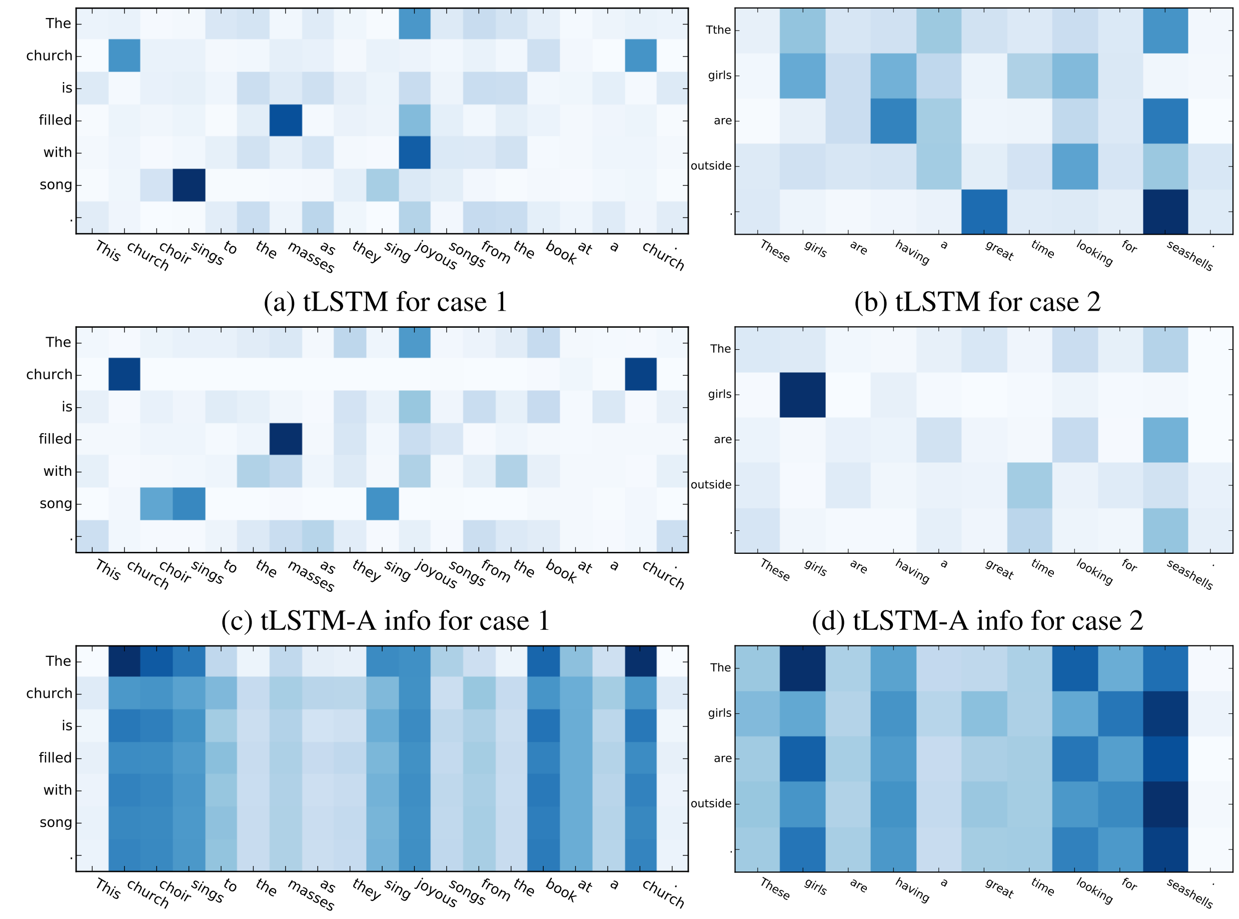
However, the intra-attention is generated just according to the information of the current sentence itself and the inter-attention focus on generating a better alignment of the two sentences without considering a better understanding of the sentence relationship. Our model intends to better understand the relation between two sentences by making the two sentence's information affect each other, which helps our model achieves good result.

4.4 Qualitative Analyses

Figure 3 listed the visualizations of the attention changing process. Figure 3a, Figure 3c, Figure 3e are the visualization of the entailment case in Table 1. Figure 3b, Figure 3d, Figure 3f are the visualization of the case in Section 1.

In Figure 3a, we can see that as the tLSTM is dealing with the hypothesis, the model pays more attention on the real meaning of the two sentences instead of simple word alignment. For example, when dealing with the word "song" in the hypothesis, the model focus more on the word "sing" in the main clause instead of the word "sing" in the subordinate clause (Figure 3c). Figure 3b has the similar phenomenon. After dealing with "outside", the model tend to focus on the words related to "outside" such as "seashells". Instead, in Figure 3d, after remove the "A info", the model cannot focus on useful information any more.

In addition, when we stop allowing the hypothesis's memory to affect the attention of the premise,



(a) tLSTM for case 1 (b) tLSTM for case 2

(c) tLSTM-A info for case 1 (d) tLSTM-A info for case 2

(e) tLSTM-C info for case 1 (f) tLSTM-C info for case 2

Figure 3: The attention visualization analysis of tLSTM, tLSTM-A info and tLSTM-C info. The premise is on the x axis, the hypothesis is on the y axis.

in Figure 3e and Figure 3f, the model's attention performs very poor. Therefore, the test accuracy of tLSTM$-$C info is much lower than tLSTM.

5 Conclusion

In this paper, we proposed a special LSTM architecture for the task of textural entailment recognizing. Inspired by the process of human reading, we bring re-read mechanic into the LSTM unit and call it re-read LSTM (rLSTM). Re-read LSTM is specially designed for dual sentence modeling and it takes the representation of the premise as general input when dealing with the hypothesis. There are two main differences between rLSTM and LSTM. First, in rLSTM, the memory of the hypothesis can affect the attention of the premise, which means the hypothesis can help the model better understanding the premise. Second, the premise's memory can affect the hidden state of the hypothesis, which means the premise can provide useful information to help the model make better decision of the relation between the premise and the hypothesis. And then, we designed an architecture for the RTE task, we use a traditional bidirectional LSTM to deal with the premise and use bidirectional rLSTM to deal with the hypothesis. Finally, the average of the bidirectional rLSTM's outputs can be used for predicting the relationship between the premise and the hypothesis.

Experiments on the SNLI corpus showed that the rLSTM model outperformed the state-of-the-art performance reported so far on this data set. Moreover, closer analyses on the attention vectors revealed that our rLSTM mechanics indeed provide and generate better attention on the premise, which represents a better understanding of the sentences.

Acknowledgements

We would like to thank our three anonymous reviewers for their helpful advice on various aspects of this work. This research was supported by the National Key Basic Research Program of China (No.2014CB340504) and the National Natural Science Foundation of China (No.61375074,61273318). The contact authors for this paper is Baobao Chang and Zhifang Sui.

References

Islam Beltagy, Cuong Chau, Gemma Boleda, Dan Garrette, Katrin Erk, and Raymond Mooney. 2013. Deep semantics with probabilistic logical form. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (* SEM-13)*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Samuel R Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. *arXiv preprint arXiv:1603.06021*.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Georgiana Dinu and Rui Wang. 2009. Inference rules and their application to recognizing textual entailment. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 211–219. Association for Computational Linguistics.

Andrew Hickl. 2008. Using discourse commitments to recognize textual entailment. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 337–344. Association for Computational Linguistics.

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

Valentin Jijkoun and Maarten de Rijke. 2005. Recognizing textual entailment using lexical similarity. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 73–76.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016a. Modelling interaction of sentence pair with coupled-lstms. *arXiv preprint arXiv:1605.05573*.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016b. Learning natural language inference using bidirectional lstm model and inner-attention. *arXiv preprint arXiv:1605.09090*.

Prodromos Malakasiotis and Ion Androutsopoulos. 2007. Learning textual entailment using svms and string similarity measures. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 42–47. Association for Computational Linguistics.

Prodromos Malakasiotis. 2011. *Paraphrase and Textual Entailment Recognition and Generation*. Ph.D. thesis, Ph. D. thesis, Department of Informatics, Athens University of Economics and Business, Greece.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 2016 Conference on Association for Computational Linguistics*, Lisbon, Portugal, August. Association for Computational Linguistics.

Rodney D Nielsen, Wayne Ward, and James H Martin. 2009. Recognizing entailment in intelligent tutoring systems. *Natural Language Engineering*, 15(04):479–501.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543.

Miguel Rios, Lucia Specia, Alexander Gelbukh, and Ruslan Mitkov. 2014. Statistical relational learning to recognise textual entailment. In *Computational Linguistics and Intelligent Text Processing*, pages 330–339. Springer.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Lei Sha, Sujian Li, Baobao Chang, Zhifang Sui, and Tingsong Jiang. 2015. Recognizing textual entailment using probabilistic inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1620–1625, Lisbon, Portugal, September. Association for Computational Linguistics.

Eyal Shnarch, Jacob Goldberger, and Ido Dagan. 2011a. A probabilistic modeling framework for lexical entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 558–563. Association for Computational Linguistics.

Eyal Shnarch, Jacob Goldberger, and Ido Dagan. 2011b. Towards a probabilistic model for lexical entailment. In *Proceedings of the TextInfer 2011 Workshop on Textual Entailment*, pages 10–19. Association for Computational Linguistics.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2015. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2006. Using dependency-based features to take the para-farce out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop*, volume 2006.

Shuohang Wang and Jing Jiang. 2015. Learning natural language inference with lstm. *arXiv preprint arXiv:1512.08849*.

Rui Wang and Günter Neumann. 2007. Recognizing textual entailment using sentence similarity based on dependency tree skeletons. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 36–41. Association for Computational Linguistics.

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*.

Fabio Massimo Zanzotto and Alessandro Moschitti. 2006. Automatic learning of textual entailments with cross-pair similarities. In *International conference on computational linguistics and the 44th Annual meeting of the Association for computational linguistics*, volume 1, pages 401–408. Association for Computational Linguistics.

Matthew D Zeiler. 2012. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.