Recognizing Textual Entailment via Multi-task Knowledge Assisted LSTM

Lei Sha, Sujian Li, Baobao Chang, Zhifang Sui

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@pku.edu.cn, {lisujian, chbb, szf}@pku.edu.cn

Abstract. Recognizing Textual Entailment (RTE) plays an important role in NLP applications like question answering, information retrieval, etc. Most previous works either use classifiers to employ elaborately designed features and lexical similarity or bring distant supervision and reasoning technique into RTE task. However, these approaches are hard to generalize due to the complexity of feature engineering and are prone to cascading errors and data sparsity problems. For alleviating the above problems, some work use LSTM-based recurrent neural network with word-by-word attention to recognize textual entailment. Nevertheless, these work did not make full use of knowledge base (KB) to help reasoning. In this paper, we propose a deep neural network architecture called **M**ulti-task **K**nowledge **A**ssisted **L**STM (MKAL), which aims to conduct implicit inference with the assistant of KB and use predicate-to-predicate attention to detect the entailment between predicates. In addition, our model applies a multi-task architecture to further improve the performance. The experimental results show that our proposed method achieves a competitive result compared to the previous work.

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 H) can be inferred (entailed) from the other one (T)[3]. 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.

Previous RTE works mainly use classifiers to employ elaborately designed features and lexical similarity. Among them, [8] break the T-H pair apart into discourse commitments and then the RTE task is reduced to the identification of the commitments from T which are most likely to support the inference of the commitments from H. However, the discourse commitments are still derived from the original text and can not present the implicit meaning latent behind the text. [18] proposed a probabilistic inference framework which transfer the discourse commitments of T-H pairs into predicate-argument structure and use distant supervision as well as Markov Logic Network (MLN) to infer the correctness of H given T. However, the performance of MLN is limited by data sparsity problem, namely, the MLN's inference rules can only cover a small portion of predicates. [17] proposed an attentive LSTM method, which use word-by-word attention to model the entailment between words. However, it did not use the KB information to help the model to infer the truth.

In this paper, we propose a deep neural network architecture called Multi-task Knowledge Assisted LSTM (MKAL) for recognizing textual entailment, which use an embedded knowledge base (KB) to help the deep neural network for implicit reasoning. We use PTransE [10], a state-of-the-art KB embedding system which can model the inference rules in KB, to calculate the embeddings of each entities and relations in the KB. We transfer the sentences into predicate-argument triples as [18] did using a convolutional neural network. And then, we input the predicates' embeddings into LSTM for implicit inference. Moreover, we add predicate-to-predicate attention into our model to detect the inference relationship between predicates. In addition, we take relation classification task as an auxiliary task to facilitate our RTE task.

The main contributions of our work are as follows:

- We propose a deep neural network architecture called Multi-task Knowledge Assisted LSTM (MKAL) for RTE task, which can integrate KB information into deep neural network;
- We propose an easy-generalized KB-assistant method, which can make full use of the potential inference rules in KB without using traditional reasoning method like MLN;
- We apply predicate-to-predicate attention to detect the inference relationship between predicates;
- We propose a multi-task architecture, which use relation classification task as an auxiliary task to help compensating the lack of supervision in our main task (RTE). The experiment result shows its superior effect.

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''))[13, 9, 21, 23, 22, 4, 14, 12]. 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, [8, 18, 20, 19, 1, 16] 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 [2, 17] start to show its effectiveness. [17] uses the attention-based technique to improve the performance of LSTM-based recurrent neural network. However, they did not take advantage of the knowledge base information for inference, so that they cannot handle the cases where the facts in H are implicitly contained by T (the facts in H cannot be directly got from the facts in T. Instead, they should be inferred by the facts in T).

3 Method

Fig 2 shows the architecture of our model. The input is the T-H sentences and the pre-identified entities. The entitiy's semantic embedding is just the word embedding of its head word. The relation's semantic embedding is calculated by convolutional neural network (CNN). For conducting implicit reasoning, KB inference information was brought into our model, namely, the entity's/relation's semantic embeddings were concatenated with their corresponding PTransE embeddings (which contains the KB information). After that, predicate-by-predicate attention based LSTM is applied to generate the representation of T-H pair. Finally, a logistic regression classifier is used to judge the entailment class (Entailment, Neutral, Contradiction) of the T-H pair. In addition, we use an auxiliary task to facilitate our main task (RTE).

3.1 Relation Embedding Calculation

In order to better represent the implicit information of the sentence, we decide to transfer the sentence into a series of predicate-argument triples "(entity A, relation, entity B)" as [18] did. We decide to calculate the relation embedding instead of extracting the relation string directly for the following two reasons: (1) The relation between two entities can be easily figured out if the origin sentence and the two entities are given; (2) The process of extracting relation string may suffer from great information loss and lacks generalization.

Inspired by [25], we use CNN to calculate the relation embedding. As is shown in Fig 1, in the Window Processing component, each token is further represented as Word Features (WF)(which is composed of the embeddings of each word) and Position Features (PF)(of the two entities). Then, the vector goes through a convolutional component. Finally, we obtain the relation embedding through a non-linear transformation.

The PF is composed of the relative distance of the current word to the head word of the two entities. For example, in the T sentence in Fig 2, the relative distances of "married" to "Senna" and "doctor" are 2 and -3, respectively. And then, the relative distances are mapped to a vector of dimension d_p (a hyperparameter) which is randomly initialized. Then we obtain the distance

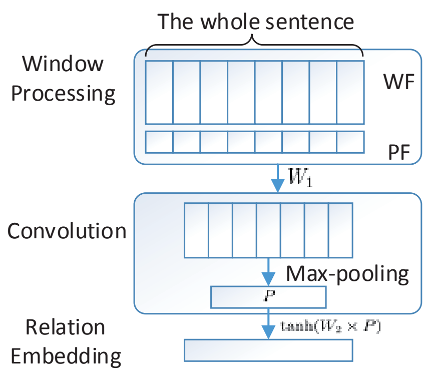

Fig. 1. The CNN architecture for relation embedding

vectors d_1 and d_2 with respect to the relative distance of the current word to the two entities. $PF = [d_1, d_2]$. Finally, the feature vector of each word is $[WF, PF]$.

The convolution operation is stated as Eq 1:

$$Z = W_1 X \tag{1}$$

where $X \in \mathbb{R}^{n_0 \times t}$ is the output of the window processing, $n_0 = w \times n$, w is the window size, n is the dimension of feature vector, and t is the token number of the input sentence. $W_1 \in \mathbb{R}^{n_1 \times n_0}$, where n_1 is the size of max-pooling layer. To determine the most useful feature in each dimension of the feature vectors, we perform max-pooling over time on the convolution result $Z \in \mathbb{R}^{n_1 \times t}$ to get the max-pooling layer P. Finally, the relation embedding is calculated as Eq 2.

$$r = \tanh(W_2 P) \tag{2}$$

where $W_2 \in \mathbb{R}^{n_2 \times n_1}$ is the linear transformation matrix, $r \in \mathbb{R}^{n_2}$ is the relation embedding.

3.2 KB Reasoning

In many RTE cases, the facts in H cannot be directly got from the facts in T. Instead, they should be inferred by the facts in T. For example, as is shown in the T-H pair of Fig 2, the fact

(Ayrton Senna, lives in, Texas)

in H should be inferred from the facts

(Ayrton Senna,married to,a doctor)
(doctor, lives in, Austin)
(Austin, is the capital of, Texas)

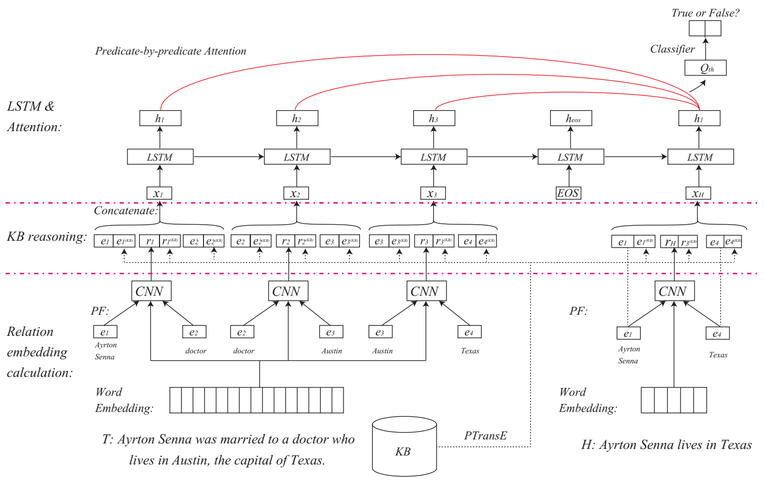
True or False?
Classifier
Q_{th}
Predicate-by-predicate Attention
LSTM & Attention:
h_1
h_2
h_3
h_{eos}
h_l
LSTM
x_1
x_2
x_3
EOS
x_H
Concatenate:
KB reasoning:
e_1 $e_1^{(KB)}$ r_1 $r_1^{(KB)}$ e_2 $e_2^{(KB)}$
e_2 $e_2^{(KB)}$ r_2 $r_2^{(KB)}$ e_3 $e_3^{(KB)}$
e_3 $e_3^{(KB)}$ r_3 $r_3^{(KB)}$ e_4 $e_4^{(KB)}$
e_1 $e_1^{(KB)}$ r_H $r_3^{(KB)}$ e_4 $e_4^{(KB)}$
CNN
PF:
Relation embedding calculation:
e_1 Ayrton Senna
e_2 doctor
e_2 doctor
e_3 Austin
e_3 Austin
e_4 Texas
Word Embedding:
T: Ayrton Senna was married to a doctor who lives in Austin, the capital of Texas.
KB
PTransE
H: Ayrton Senna lives in Texas

Fig. 2. The Main Framework of MKAL

in T. So we need inference rules derived by KB to help us conduct reasoning.

Since the inference rules are in the form: $r_1 \wedge r_2 \Rightarrow r_3$, which is just the same form as modeled by PTransE [10], which is a KB embedding method modeling relation paths like $r_1 + r_2 = r_3$. Therefore, the representations of entities and relations learned by PTransE contain the information of implicit inference rules. So we use PTransE to model the inference rules in KB. We take the entities' and relations' PTransE embeddings as well as the semantic embeddings together as the input of the LSTM-RNN to conduct inference as is shown in Fig 2.

We use a heuristic method to find the corresponding KB items for the entities and relations.

For entities, we first compare the surface entity string with the KB entities. If the entity can be found in KB, we will use the corresponding PTransE embedding as this entity's KB information. If the entity cannot be found in KB, then KB cannot bring it any information, so we just set its' KB information as zero.

Since we do not hope to explicitly extract the relation string from text (a perfect textual relation is too hard to extract), but we do hope to find a match for the textual relations from KB relations. So we use a syntactic constraint method [5] to extract **raw** relation string (RRS) from two adjacent entities. The syntactic constraint requires relation phrase to match the POS tag pattern shown in Figure 3. The pattern limits relation phrases to be either a simple verb phrase (e.g., invented), a verb phrase followed immediately by a preposition or particle (e.g., located in), or a verb phrase followed by a simple noun phrase and ending in a preposition or particle (e.g., has atomic weight of). Different from [5], we also allow the case of only one preposition (e.g. from, at), because this kind of relation usually states for location and affiliation.

When matching the KB relations and the extracted RRS, we calculate their semantic embedding in prior by averaging the embedding of each word. We choose the KB relation which has the minimal cosine distance to the extracted RRS as its corresponding KB relation.

Compared to [18], which use AMIE [6] to extract inference rules from KB and then use MLN to conduct inference, our method can cover all entities and relations in KB, which may alleviate the data sparsity problem.

$V \vert VP \vert V^{*}W^{*}P$
V = verb particle? adv?
W = (noun|adj | adv | pron | det)
P = (prep|particle | inf. marker)

Fig. 3. A simple POS-based regular expression

3.3 LSTM-based Attentive Neural Reasoner

LSTM 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 [2]. LSTMs use memory cells to store information for a long period. In our model, the better LSTM remembers the predicates, the more accurate recognizing result it can give. Given 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 LSTM transition equations are listed in Eq 3, $\odot$ is element-wise multiplication.

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) & \widetilde{C}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) & c_t &= i_t \odot \widetilde{C}_t + f_t \odot c_{t-1} \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) & h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{3}$$

Attentive LSTM for Predicate Reasoning As is shown in Fig 2, we take the combination of semantic embedding and PTransE embedding of predicate-argument triples as input to LSTM.

For determining the entailment of individual predicates, we use predicate-to-predicate attention in our model, which can model the inference relationship between the predicates in T and H. Following [17], we attend over the first LSTM's output vectors of T, while the second LSTM processes H one predicate at a time. Denote $Y \in \mathbb{R}^{d \times L}$ as a matrix consisting of the T sentence's LSTM output vector $[h_1 \cdots h_L]$, this can be modeled as follows:

$$\begin{aligned} M_t &= \tanh(W_y Y + (W_h h_t + W_r r_{t-1}) \otimes e_L) \\ \alpha_t &= softmax(w^T M_t) \\ r_t &= Y \alpha_t^T + \tanh(W^t r_{t-1}) \end{aligned} \tag{4}$$

where $M_t \in \mathbb{R}^{d \times L}$, $\alpha_t \in \mathbb{R}^d$ is the attention weight vector over all output vectors of T for every predicate x_t with $t \in (L+1, N)$ in H and $r_t \in \mathbb{R}^d$ is dependent on the previous attention representation r_{t-1} to inform the model about what was attended over in the previous step. The outer product $\otimes$ is to repeat the previous operand as many times as the word number in T (L times).

The final T-H pair representation is Eq 5, which is obtained from a non-linear combination of the last attention-weighted representation r_N and the last LSTM output vector h_N.

$$h^* = \tanh(W^p r_N + W^x h_N) \tag{5}$$

where $h^* \in \mathbb{R}^d$. Then h^* is fed into a classifier:

$$O = W^* h^* + b^* \tag{6}$$

where $O \in \mathbb{R}^3$ is the final output of this T-H pair, each entry contains the score of an entailment class (entailment, neutral, contradiction).

3.4 Multi-task Training for Relation Representation

Multi-task learning (MTL) is a kind of machine learning approach, which trains both the main task and auxiliary tasks simultaneously with a shared representation learning the commonality among the tasks. In our work, we use auxiliary training to compensate the lack of supervision in the main task. Intuitively, the CNN which is used to calculate relation embeddings can be improved by relation classification task (RC). Therefore, we use the relation classification task to assist our main task (RTE).

The architecture of our multi-task neural network is shown in Fig 4. While the relation semantic embeddings calculated by CNN as well as PTransE embeddings were used for LSTM, the relation semantic embedding were also taken as the feature vector of the RC task as shown in Eq 7. Given an input example x, the CNN outputs the vector O_a, the conditional probability $p(i|x, \theta_a, \theta_s)$ for the i-th component is calculated by softmax operation.

$$\begin{aligned} O_a &= W_3 r \\ p(i|x, \theta_a, \theta_s) &= \frac{O_a^{(i)}}{\sum_k O_a^{(k)}} \end{aligned} \tag{7}$$

where $\theta_a = \{W_3\}$ represents for the RC-task-only parameters, $\theta_s = \{W_1, W_2\}$ represents for the shared parameters. Given all the training examples $x_a^{(i)}, y_a^{(i)}$ of relation classification, the loss function of this auxiliary task is shown as Eq 8.

$$J_a(\theta_a, \theta_s) = \sum_i \log p(y_a^{(i)}|x_a^{(i)}, \theta_a, \theta_s) \tag{8}$$

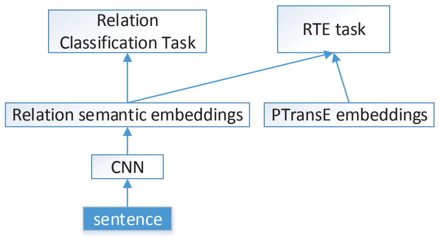

Fig. 4. Multi-task Architecture. "SE" represents sentence embedding

3.5 Model Training

We define the ground-truth label vector y for each T-H pair as a binary vector. If this T-H 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 LSTM neural network parameters θ_{RTE}, the objective function for a T-H pair can be written as,

$$J(\theta_{RTE}, \theta_s) = -\sum_i y(i)\log(O(i)) + \frac{\lambda_1}{2}\|\theta_{RTE}\|^2 + \frac{\lambda_2}{2}\|\theta_s\|^2 \tag{9}$$

We use mini-batch AdaDelta [24] to train the parameters $\theta_{RTE}, \theta_s, \theta_a$. Referring to the training procedure in [11], we select one task in each epoch and update the model according to its task-specific objective (J or J_a).

We expect the two tasks to reach their best performance at nearly the same time. Therefore, we assign different regulative ratio μ and μ_a to different tasks to adjust the learning rate of AdaDelta.

4 Experiments

4.1 Datasets and Model Configuration

We conduct experiments on the Stanford Natural Language Inference corpus (SNLI) [2]. Since our main task is RTE, Table 1 summarizes the statistics of the three entailment classes in SNLI.

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

Table 1. Distribution of Entailment Classes in SNLI

For the auxiliary task (RC task), we use the SemEval-2010 Task 8 dataset [7], which contains 10717 annotated examples, including 8000 training instances and 2717 test instances.

For each T-H pair, we extract the noun phrase (NP) in prior as entities by extracting each NP node in the sentence's dependency parse tree. We use the pre-trained word embeddings provided by GloVe [15], and the dimension of the embeddings d is 50. The hyperparameters of our model are set as in Table 2.

Word embedding size	$d = 50$
Regularization	$\lambda_1 = 0.004$ $\lambda_2 = 0.001$
regulative ratio	$\mu = 1.0$ $\mu_a = 0.5$
Dropout fraction	0.5
Embedding Length (Entity & Relation)	$d_e = 50$
PTransE Embedding Length (Entity & Relation)	$d_e^{(KB)} = 50$
CNN hyperparameters	$w = 3$ $d_p = 10$ $n = d + 2d_p$ $n_1 = 200$ $n_2 = 100$

Table 2. Hyperparameters of our model

4.2 Overall Performance

Method	Accuracy
LSTM[2]	77.6
Classifier[2]	78.2
Attention[17]	83.5
MKAL(CNN+LSTM)	78.9
+ Multi-task	82.5
+ KB	83.8
+ Attention	84.2

Table 3. The performance on SNLI test set

We compare our result with the following state-of-art methods:

- **LSTM[2]** is a LSTM-based method which encodes T and H independently. The encodings of T and H are concatenated and fed into a deep neural network classifier.

– **Classifier**[2] is a feature-engineered classifier which use a large set of elaborately designed features to recognize the entailment of T and H.
– **Attention**[17] is a LSTM method which processes H sentence conditioned on T sentence while using attention technique.

Table 3 shows the performance on SNLI dataset of our model. For clarity, we split our result into four parts. MKAL(CNN+LSTM) is the most basic model, which only use CNN to calculate the relation embedding and then input all the triples'(entity A, relation, entity B) embeddings into LSTM encoder. Finally, the output of LSTM is taken as the feature representation of the T-H pair. We can see that MKAL(CNN+LSTM) has achieved an accuracy of 78.9%, which cannot outperform the state-of-art result of [17]. One possible reason is that the process of using CNN to extract the relations' representation may lead to severe information loss. After we add an auxiliary task (RC task), the multi-task architecture makes the RTE accuracy improved to 82.5%, which is a great improvement. Then we add PTransE embedding (+ KB) to bring inference information to our model, which improves the accuracy to 83.8%. Finally, the attention technique (+ Attention) has improved the accuracy to 84.2%, which has outperformed the state-of-the-art results (Wilcoxon signed-rank test, $p < 0.05$).

4.3 Effect of Multi-Task

Method	F_1
CNN[25]	82.7
Our Auxiliary task	80.4

Table 4. The comparison of relation classification task performance between our auxiliary task and previous works

According to Table 3, the overall performance can be greatly improved by the auxiliary task. The relation classification task's performance is shown in Table 4. Although our auxiliary task did not outperform the origin CNN model in [25], the auxiliary task is comparable to the origin CNN model. The reasons why we did not achieve a better result in the auxiliary task are as follows: First, we did not use the lexical features described in [25] for simplicity. Second, the supervise information of RTE may be not enough for the relation classification task.

4.4 Effect of KB

Of all the extracted RRS in the SNLI test set, about 19.6% can be matched with KB relations in YAGO. For the entities in the SNLI test set, about 5.4% can be matched with YAGO's entities. Although the ratio of matched entities/relations is not very high, the KB information is still effective to the overall performance (1.3 percentage point according to Table 3). Some examples of matched entities and relations are shown in Table 5.

Entity	KB Entity
a church	⟨Vowchurch⟩
Two women	⟨Two Women⟩
the theater	⟨Guthrie Theater⟩

RRS	KB Relation
connected to	⟨isConnectedTo⟩
directs	⟨directed⟩
plays	⟨playsFor⟩

Table 5. Example of matched entities and relations

4.5 Effect of Attention

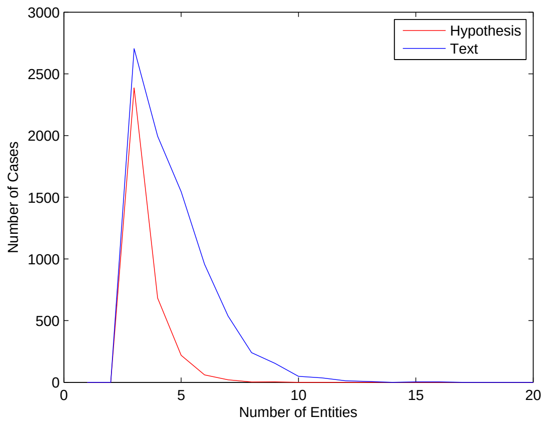

Fig. 5. The distribution of entity numbers of each T-H pair

Making the model enable to attend over output vectors of T for every predicate in H gains another 0.4 percentage point improvement. We argue that this is due to the model being able to check for the entailment between the predicates in T and H.

Visualizations of predicate-by-predicate attention are depicted in Fig 6. The original sentence of Fig 6's left figure is as follows:

T: This church choir sings to the masses as they sing joyous songs from the book at a church.
H: The church is filled with song.

According to the figure, the predicate "singto (church choir,mass)", "sing (mass,song)" and "at (book,church)" are important for entailing the H predicate "Fillwith (church,song)", which is reasonable for human. The original sentence of Fig 6's right figure in Fig 6 is the example in Fig 2. In this attention matrix, we can see that all of the three predicates in T are important to entail the H's predicate.

However, the attention technique did not bring too much improvement compared to other works [17]. One possible reason is, the size of our attention matrix is much smaller than other works since we only use predicate-by-predicate attention. There may not be too much predicates in a sentence, but a sentence usually contains many words. In our model, we define each two adjacent entities and their relation make a predicate. Statistically, in Fig 5, we illustrate the distribution of the number of entities in one sentence. We can see that most of the T sentences have less than 10 entities, which means less than 9 predicates; most of H sentences have less than 5 entities, which means less than 4 predicates. Smaller size may weaken the role of attention. For example, in Fig 6's right figure, all of the three predicates are attended, so that it may not make too much difference than there is no attention technique at all. Therefore the effect of attention matrix is weaker than the effect in previous works.

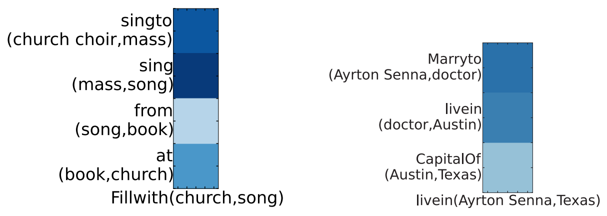

Fig. 6. Predicate-by-predicate attention visualization. Darker-color block represents more attention between the two predicates

5 Conclusion

In this paper, we use a multi-task architecture to recognize textual entailment. We use relation classification task to facilitate the recognizing textual entailment (RTE) task (main task). The main task use CNN to calculate the relations' semantic embeddings and concatenate the adjacent entities' embedding and their relation's embedding as the predicate's semantic embedding. At the same time, we give each entity and relation a PTransE embedding (KB information), and put all of them into LSTM-based recurrent neural network so that implicit reasoning can be conducted. With the help of predicate-by-predicate attention technique, we detect the entailment between predicates. The experiments show that the KB information, the attention technique, especially the multi-task architecture are effective to the RTE task.

References

1. Beltagy, I., Chau, C., Boleda, G., Garrette, D., Erk, K., Mooney, R.: Deep semantics with probabilistic logical form. In: Proceedings of the Second Joint Conference on Lexical and Computational Semantics (* SEM-13) (2013)
2. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. arXiv preprint arXiv:1508.05326 (2015)
3. Dagan, I., Glickman, O., Magnini, B.: The pascal recognising textual entailment challenge. In: Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment, pp. 177–190. Springer (2006)
4. Dinu, G., Wang, R.: 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. pp. 211–219. Association for Computational Linguistics (2009)
5. Etzioni, O., Fader, A., Christensen, J., Soderland, S., Mausam, M.: Open information extraction: The second generation. In: IJCAI. vol. 11, pp. 3–10 (2011)
6. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd international conference on World Wide Web. pp. 413–422. International World Wide Web Conferences Steering Committee (2013)
7. Hendrickx, I., Kim, S.N., Kozareva, Z., Nakov, P., Ó Séaghdha, D., Padó, S., Pennacchiotti, M., Romano, L., Szpakowicz, S.: Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In: Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions. pp. 94–99. Association for Computational Linguistics (2009)
8. Hickl, A.: Using discourse commitments to recognize textual entailment. In: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. pp. 337–344. Association for Computational Linguistics (2008)
9. Jijkoun, V., de Rijke, M.: Recognizing textual entailment using lexical similarity. In: Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment. pp. 73–76 (2005)
10. Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., Liu, S.: Modeling relation paths for representation learning of knowledge bases. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 705–714. Association for Computational Linguistics, Lisbon, Portugal (September 2015), https://aclweb.org/anthology/D/D15/D15-1082
11. Liu, X., Gao, J., He, X., Deng, L., Duh, K., Wang, Y.Y.: Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 912–921. Association for Computational Linguistics, Denver, Colorado (May–June 2015), http://www.aclweb.org/anthology/N15-1092
12. Malakasiotis, P.: Paraphrase and Textual Entailment Recognition and Generation. Ph.D. thesis, Ph. D. thesis, Department of Informatics, Athens University of Economics and Business, Greece (2011)
13. Malakasiotis, P., Androutsopoulos, I.: Learning textual entailment using svms and string similarity measures. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. pp. 42–47. Association for Computational Linguistics (2007)

14. Nielsen, R.D., Ward, W., Martin, J.H.: Recognizing entailment in intelligent tutoring systems. Natural Language Engineering 15(04), 479–501 (2009)
15. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) 12, 1532–1543 (2014)
16. Rios, M., Specia, L., Gelbukh, A., Mitkov, R.: Statistical relational learning to recognise textual entailment. In: Computational Linguistics and Intelligent Text Processing, pp. 330–339. Springer (2014)
17. Rocktäschel, T., Grefenstette, E., Hermann, K.M., Kočiskỳ, T., Blunsom, P.: Reasoning about entailment with neural attention. arXiv preprint arXiv:1509.06664 (2015)
18. Sha, L., Li, S., Chang, B., Sui, Z., Jiang, T.: Recognizing textual entailment using probabilistic inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1620–1625. Association for Computational Linguistics, Lisbon, Portugal (September 2015), https://aclweb.org/anthology/D/D15/D15-1185
19. Shnarch, E., Goldberger, J., Dagan, I.: 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. pp. 558–563. Association for Computational Linguistics (2011)
20. Shnarch, E., Goldberger, J., Dagan, I.: Towards a probabilistic model for lexical entailment. In: Proceedings of the TextInfer 2011 Workshop on Textual Entailment. pp. 10–19. Association for Computational Linguistics (2011)
21. Wan, S., Dras, M., Dale, R., Paris, C.: Using dependency-based features to take the para-farce out of paraphrase. In: Proceedings of the Australasian Language Technology Workshop. vol. 2006 (2006)
22. Wang, R., Neumann, G.: Recognizing textual entailment using sentence similarity based on dependency tree skeletons. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. pp. 36–41. Association for Computational Linguistics (2007)
23. Zanzotto, F.M., Moschitti, A.: 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. vol. 1, pp. 401–408. Association for Computational Linguistics (2006)
24. Zeiler, M.D.: Adadelta: An adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
25. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation classification via convolutional deep neural network. In: Proceedings of COLING. pp. 2335–2344 (2014)