Adopting the Word-Pair-Dependency-Triplets with Individual Comparison for Natural Language Inference

Qianlong Du[1,2], Chengqing Zong[1,2,3], Keh-Yih Su[4]
[1]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Science
[2]University of Chinese Academy of Science
[3]CAS Center for Excellence in Brain Science and Intelligence Technology
[4]Institute of Information Science, Academia Sinica
{qianlong.du, cqzong}@nlpr.ia.ac.cn
kysu@iis.sinica.edu.tw

Abstract

This paper proposes to perform natural language inference with *Word-Pair-Dependency-Triplets*. Most previous DNN-based approaches either ignore syntactic dependency among words, or directly use tree-LSTM to generate sentence representation with irrelevant information. To overcome the problems mentioned above, we adopt *Word-Pair-Dependency-Triplets* to improve alignment and inference judgment. To be specific, instead of comparing each triplet from one passage with the merged information of another passage, we first propose to perform comparison directly between the triplets of the given passage-pair to make the judgment more interpretable. Experimental results show that the performance of our approach is better than most of the approaches that use tree structures, and is comparable to other state-of-the-art approaches.

1 Introduction

Natural language inference (NLI) refers to the following task: given a text passage *P* (*Premise*) (which might have more than one sentence) and a text passage *H* (*Hypothesis*), whether we can infer *H* from *P*, i.e., identifying a specific relationship among *entailment*, *neutral* and *contradiction*. It has many applications such as question answering (Bhaskar et al., 2013; Harabagiu and Hickl, 2006), information extraction (Romano et al., 2006), machine translation (Pado et al., 2009), automatic text summarization (Harabagiu et al., 2007) and so on. Some evaluations about this task have been organized in the past decades, such as the PASCAL Recognizing Textual Entailment (RTE) Challenge (Dagan et al., 2005), SemEval-2014 (Marelli et al., 2014) and RITE (Shima et al., 2011).

Many previous approaches adopt statistical frameworks (Heilman et al., 2010; Kouylekov and Magnini, 2005). However, neural network approaches have emerged after Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015) was released. Most of them adopt an increasingly complicated network structure to represent text passages, and then predict the relationship between them (Bowman et al., 2016; Liu et al., 2016b). However, *P* might include extra words which are not directly related to *H*. Actually, only the words in *P* that are associated with the words in *H* should be paid attention to. Those relevant words should be emphasized more while the irrelevant words should be less weighted during decision making. Therefore, some approaches (Parikh et al., 2016; Chen et al., 2017a) adopt attention mechanism to implicitly align the words between two passages to yield a better performance. This idea is very similar to how human make the entailment judgment, and the result shows that it is very effective for performing natural language inference on SNLI corpus in which most words in *H* can find their corresponding ones in *P*.

Proceedings of the 27th International Conference on Computational Linguistics, pages 414–425
Santa Fe, New Mexico, USA, August 20-26, 2018.

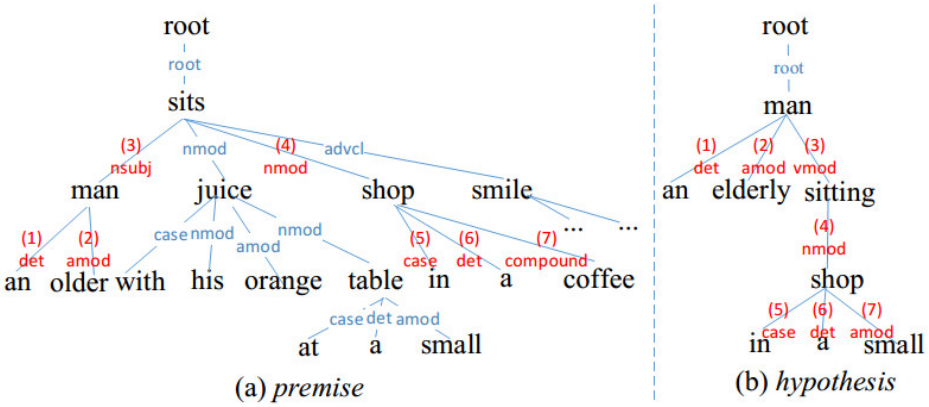

Figure 1. Dependency trees[1] for **Premise** "*An older man sits with his orange juice at a small table in a coffee shop while employees in bright colored shirts smile in the background."* and **Hypothesis** *"An elderly man sitting in a small shop.*" The relationship between them is "*neutral*".

However, after having analyzed some errors generated from an attention-based approach (Parikh et al., 2016), we find that it will introduce mis-alignment and might cause wrong inference. Although context information is used to alleviate this problem, it still cannot handle long distance dependency. Take the following sentence pair as an example (The benchmark is *neutral*):

Premise: *An older man sits with his orange juice at a small table in a coffee shop while employees in bright colored shirts smile in the background.*

Hypothesis: *An elderly man sitting in a small shop.*

The attention-based approach (Parikh et al., 2016) cannot catch the relation between "shop" and "small" in P precisely (which is important for the prediction). In this example, the relationship between *P* and *H* will be predicted as "*entailment*", because all the words in *H* can be found in *P*, although the word "*small*" in these two sentences does not modify the same thing (e.g., "*small*" modifies "*table*" in *premise* while modifies "*shop*" in *hypothesis*).

The above example clearly shows that a sentence is not a set of independent words. It is a sequence of words with syntactic relationship. Based on this observation, we propose to adopt the *Word-Pair Relation-Head-Dependent* (*RHD*) triplet[2] for conducting alignment and comparison. Furthermore, Parikh et al. (2016) and other previous models only compare each word in *H* with the vector merged from all the words in *P* according to their associated alignment scores, and vice versa. However, as shown in Figure 1, human compares *H* and *P* mainly based on *Structure Analogy* (Du et al., 2016; Gentner 1983; Gentner & Markman, 1997) instead of the merged meaning which will not only import irrelevant text but also lose information during merging. Consequently, only words with closely related syntactic/semantic roles (of the aligned predicates) should be compared.

Therefore, we first create two sets of *RHD* from *P* and *H* to denote their corresponding structures, and then perform comparison between triplets in *P* and those in *H*. Accordingly, two *RHD* triplets should be aligned if their relations are related and their head-words are aligned (e.g., the triplets with the same indexes are aligned in Figure 1). Particularly, when two *RHD* triplets are compared, each part of *RHD* triplet (i.e., *Relation, Head*, and *Dependent*) should be compared separately. Besides, as the words of some triplet pairs possess reversed head-dependent relations (e.g., the *Dependent* in "(*nsubj, sits, man*)" is linked to the *Head* in "(*vmod, man, sitting*)", as shown by the triplet pair with index 3 in Figure 1), we introduce *cross-comparison* to compare the *Head* from "(*nsubj, sits, man*)" with the *Dependent* from "(*vmod, man, sitting*)", and vice versa.

[1] The words in black represent the nodes of the dependency tree, and the string on each line represents the dependency relation between two nodes. The red relation denotes that its associated triplet is important in making the judgment. Links with the same indexes indicates that they are aligned and compared when judged the result by human.

[2] Each *RHD* triplet is denoted by "(*rel, head, dep*)", where *head* and *dep* denote the *head-word* and the *dependent-word* of the dependency relation, respectively; and *rel* denotes the dependency relation between *head* and *dependent*.

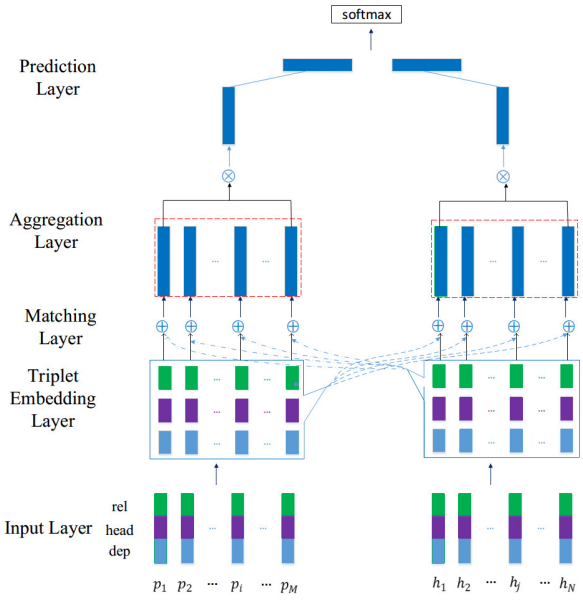

Figure 2: The skeleton of the proposed approach. The *rel*, *head* and *dep* of the triplet are represented with green, purple, and light-blue colors, respectively. Also, $\oplus$ denotes the *Comparison* operation between triplets (see section 2.2.2 for "Matching layer"), while $\otimes$ denotes the *Aggregation* operation (see section 2.2.3 for "Aggregation layer"). Besides, the left part and the right part in *Matching Layer* represent *P-aligned-to-H* and *H-aligned-to-P*, respectively.

Our contributions are summarized as follows: (1) The *RHD* triplet is first proposed to be the alignment and comparison unit in the neural network for NLI. In this way, the corresponding words could be aligned and compared more precisely. (2) Instead of comparing one *RHD* triplet of *H* with the merged meaning of all the *RHD* triplets in *P* (and vice versa), we propose to directly compare each *RHD* triplet of *H* with another *RHD* triplet of *P*; and each part in *RHD* triplet is compared separately. (3) We propose to use *cross-comparison* to compare the related words with different syntactic/semantic roles.

2 Proposed Approach

In our model, we first transform *P* and *H* into two sets of *RHD* triplets. For each *RHD* triplet of *P*, we compare it with another *RHD* triplet of *H* (without merging) to generate an *individual comparison vector* (and vice versa). Afterwards, we use a *self-attention* mechanism to align and sum them to yield the *one-side merged comparison vector* between a triplet and the triplet set of the other side. Last, we aggregate those *one-side merged comparison vectors* to give the overall entailment judgment.

Figure 2 shows the skeleton of the proposed approach. It consists of the following 5 layers: (1) *Input Layer*, which initializes the embedding of the words and relations; (2) *Triplet Embedding Layer*, which is used to adapt the input embedding to yield a better representation for this task; (3) *Matching Layer*, which performs comparison within each *RHD* triplet pair, scores the alignment weights, and sum those *individual comparison vectors* to generate the *one-side merged comparison vector* between each triplet with the triplet set of the other side; (4) *Aggregation Layer*, which aggregates the *one-side merged comparison vectors* to get a *directional comparison vector* for each comparing direction; and (5) *Prediction Layer*, which uses two separated *directional comparison vectors* and a feed-forward neural network classifier to predict the overall judgment.

Our model is symmetric about *P* and *H*. So for simplicity, we only describe the left parts which mainly about comparing each unit of *P* with *H*. Right part is exactly the same except that the roles of *P* and *H* are exchanged.

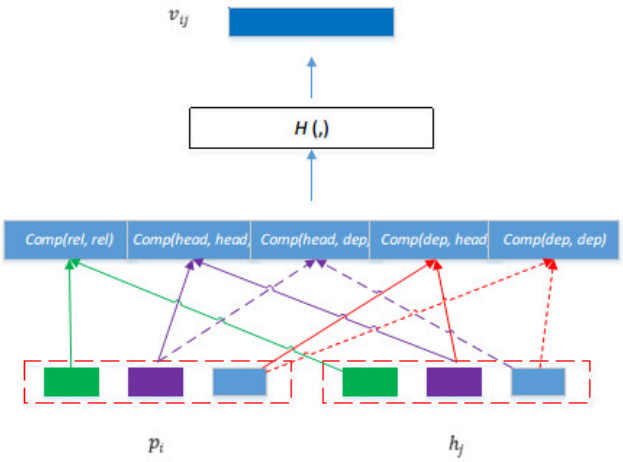

Figure 3: An illustration for the comparison between a triplet p_i in *Premise* and a triplet h_j in *Hypothesis*. The *rel*, *head* and *dep* of the triplet are represented with green, purple, and light-blue colors, respectively. *Comp(,)* indicate the comparison function denoted in equation (2). *H*(,) is a multilayer perception denoted in equation (4). The green solid-line, purple solid-line and red solid-line represent the comparison of pair (*rel, rel*), (*head, head*) and (*dep, dep*), respectively. The purple dot-line represents the cross-comparison of pair (*head, dep*), and red dot-line represents the cross-comparison of pair (*dep, head*).

2.1 Input Layer Generation

We first use a dependency parser[3] to transform *P* and *H* into two sets of *RHD* triplets. We define $\hat{P} := (p_1, p_2, \cdots, p_\mathrm{m})$ and $\hat{H} := (h_1, h_2, \cdots, h_\mathrm{n})$ be two sets of *RHD* triplets, while p_i and h_j denote the i^{th} *RHD* triplet and the j^{th} *RHD* triplet in *P* and *H*, respectively; also, *m* and *n* indicate the number of associated triplets in *P* and *H*, respectively. We then instantiate the *Input Layer* with the corresponding *word-embeddings* and *rel-embedding* of *RHD* triplets (For conciseness, we will let *rel/head/dep* denote both the original meaning and the corresponding embedding interchangeably from now on). Each $head, dep \in R^{d_w}$ is a word embedding of dimension d_w which is initialized with pre-trained GloVe word embedding (Pennington et al., 2014), while $rel \in R^{d_r}$ is a relation embedding vector of dimension d_r and is initialized randomly with a standard normal distribution (Please note, only rel will be tuned later during training). Each triplet-embedding will be presented as a triplet which contains three embedding corresponding to *rel*, *head* and *dep* respectively.

2.2 Network Architecture

2.2.1 Triplet Embedding Layer

As we fix the value of word embedding during training, in order to obtain better relation/word embedding representations to compare for this task, we use a simple feed-forward structure to adapt the three parts of the triplet to the task. The computations are as follows:

$$\begin{aligned} rel &= W_r * rel_{in} + b_r \\ head &= W_w * head_{in} + b_w \\ dep &= W_w * dep_{in} + b_w \end{aligned} \tag{1}$$

where * is the multiplication of matrices, $rel_{in} \in R^{d_r}$, $head_{in} \in R^{d_w}$ and $dep_{in} \in R^{d_w}$ are the input embedding-vectors from *Input Layer*, $rel \in R^{d_r}$, $head \in R^{d_w}$ and $dep \in R^{d_w}$ are the new representations generated, $W_r \in R^{d_r \times d_r}$, $W_w \in R^{d_w \times d_w}$, $b_r \in R^{d_r}$, $b_w \in R^{d_w}$ are the weight matrices to be learned. Here, all the *rel* share the same weight matrices, while all the words share the same weight matrices.

[3] http://nlp.stanford.edu/software/lex-parser.shtml#Download

2.2.2 Matching Layer

This layer is the core of our model. It is mainly used to perform the individual comparison between two triplets and use associated alignment weights to focus on the individual-comparisons of the preferred alignments. In this step, we will use a *one-side merged comparison vector* to represent each comparison result of one triplet with the triplet-set of another side. For a triplet p_i from *Triplet Embedding Layer*, it has three parts: $rel_{p_i} \in R^{d_r}$, $head_{p_i} \in R^{d_w}$ and $dep_{p_i} \in R^{d_w}$; and for h_j, it has $rel_{h_j} \in R^{d_r}$, $head_{h_j} \in R^{d_w}$ and $dep_{h_j} \in R^{d_w}$.

Figure 3 shows how the *individual comparison-vector* of one triplet-pair is generated. The vector $v_{ij} \in R^d$ denotes the comparison result between triplet p_i and a triplet h_j from $\hat{H}$, while d is the dimension of hidden layer. During comparison, each component of the triplet (i.e., *rel*, *head* and *dep*) is compared independently, as shown in the figure. Here, the comparing function is denoted as *comp(,)* in equation (2). G is a multi-layer perceptron with one hidden layer and a *Relu* activation.

$$comp(v_1, v_2) = G([v_1; v_2; v_1 - v_2; v_1 \odot v_2]) \tag{2}$$

Where v_1 and v_2 are any two embedding vectors. The notation ";" (within the bracket-pair in the above equation) denotes *Concatenation*; also, '$-$' and '$\odot$' are the *difference* and *element-wise product* of vectors respectively. Then we can get the comparison results in Figure 3 as follows:

$$\begin{aligned}
\overline{rel_{ij}} &:= comp\left(rel_{p_i}, rel_{h_j}\right) \\
\overline{head_{ij}} &:= comp\left(head_{p_i}, head_{h_j}\right) \\
\overline{head_x_{ij}} &:= comp\left(head_{p_i}, dep_{h_j}\right) \\
\overline{dep_{ij}} &:= comp\left(dep_{p_i}, dep_{h_j}\right) \\
\overline{dep_x_{ij}} &:= comp\left(dep_{p_i}, head_{h_j}\right)
\end{aligned} \tag{3}$$

Where $\overline{head\_x_{ij}}$ and $\overline{dep\_x_{ij}}$ are the results of *cross-comparison*. Please note that the comparison functions with the same input arguments share the same set of parameters in G. For example, all the functions *comp*(*head*, *head*) share a set of parameters while all *comp*(*head*, *dep*) share another set of parameters. After we obtain the comparison results of these components, we can incorporate them to yield the triplet *individual comparison vector* v_{ij} between p_i and h_j as follow.

$$v_{ij} = H([\overline{rel_{ij}}; \overline{head_{ij}}; \overline{head\_x_{ij}}; \overline{dep_{ij}}; \overline{dep\_x_{ij}}]) \tag{4}$$

Here H is a multi-layer perceptron with two hidden layers and a *Relu* activation. Afterwards, we need to generate the *alignment weight* e_{ij} between triplet p_i and triplet h_j to extract the key information for judgment. Most of the previous models (Chen et al., 2017a; Parikh et al., 2016) use the multiplication of two semantic unit vectors as their alignment weights. However, we find the individual comparison here can describe the relatedness of those triplet-pairs better. Consequently, we generate the *alignment weights* using these *individual comparison vectors* with a self-attention mechanism as follows.

$$e_{ij} = W_{s2}\tanh(W_{s1}v_{ij}) \tag{5}$$

Where $W_{s1} \in R^{d \times d}$ and $W_{s2} \in R^{d \times 1}$ are weight matrices to be learned. Then we use e_{ij} to obtain the *one-side merged comparison vectors* $O_{i,H}$ from various v_{ij} as follow.

$$O_{i,H} = \sum_{j=1}^{l_h} \frac{exp(e_{ij})}{\sum_{k=1}^{l_h} exp(e_{ik}))} v_{ij} \tag{6}$$

Where $O_{i,H}$ is the *one-side merged comparison vector* between p_i and the whole set H, l_h is the number of *RHD* triplets in H; $O_{P,j}$ (the right part in Figure 2) is defined similarly between h_j and the whole P.

2.2.3 Aggregation Layer

In this layer, we aggregate all the *one-side merged comparison vectors* $O_{i,H}$ and $O_{P,j}$ (obtained above) to generate the final comparison vector for these two different directions between P and H. Like the previous approaches (Chen et al., 2017a; Parikh et al., 2016), we aggregate the information by summation and max pooling:

$$\begin{aligned} O_{P,sum} &= \sum_{i=1}^{l_p} O_{i,H}, \qquad O_{P,max} = \max_{i=1}^{l_p} O_{i,H}, \qquad O_P = [O_{P,sum}; O_{P,max}] \\ O_{H,sum} &= \sum_{j=1}^{l_h} O_{P,j}, \qquad O_{H,max} = \max_{j=1}^{l_h} O_{P,j}, \qquad O_H = [O_{H,sum}; O_{H,max}] \end{aligned} \tag{7}$$

Where l_p and l_h are the numbers of triplets in P and H, respectively. We get the *overall comparison vector* O_P by concatenating the summation $O_{P,sum}$ and the max pooling $O_{P,max}$ using the *one-side merged comparison vectors of P* (and vice versa). Afterwards, we use these *overall comparison vectors* to predict the relationship in the next section.

2.2.4 Prediction Layer

From the above section, we have obtained O_P and O_H, which are the *overall comparison vectors* from $\hat{P}$ to $\hat{H}$ and from $\hat{H}$ to $\hat{P}$, respectively (i.e., the *overall comparisons* in two different directions). We then concatenate them and use a multi-layer perceptron classifier Q (it has two hidden layers with *Relu* activation and a *softmax* output layer) to generate the final overall judgment vector $\hat{y}$ (as shown in Eq. (8)), where $\hat{y} \in R^C$ (C equals the number of classes) are the scores for each class. The predicted class can be got by setting $y = \text{argmax}_i \hat{y}_i$. When we train the model, we use multi-class cross-entropy loss with dropout regularization (Srivastava et al., 2014).

$$\hat{y} = Q([o_P; o_H]) \tag{8}$$

3 Experiments

3.1 Dataset

We adopt both *SNLI* (Bowman et al., 2015) corpus[4] and *MultiNLI* (Williams et al., 2018) corpus[5] to test the performance. They are briefly introduced as follows.

SNLI - It contains 570k sentence pairs. The sentence pairs in this corpus are labelled with one of the following relationships: *entailment*, *contradiction*, *neutral* and "-", where "-" means that it lacks of consensus from human annotators. In our experiments, we follow Bowman et al. (2015) to delete those sentence pairs labelled with "-". Consequently, we end up with 549,367 pairs for training, 9,842 pairs for development and 9,824 pairs for testing.

MultiNLI - This corpus has 433k sentence pairs, which are collected from broad range of genre of American English such as written non-fiction genres (e.g. SLATE, OUP), spoken genres (TELEPHONE, FACE-TO-FACE), less formal written genres (FICTION, LETTERS), and that specialized on for 9/11. For the training set of this corpus, it selects half of the genres to create in-domain (matched) and out-domain (mismatched) development/test sets. Since the test set labels of this corpus are not released, the test performance is obtained through submission to Kaggle.com[6].

[4] https://nlp.stanford.edu/projects/snli/
[5] http://www.nyu.edu/projects/bowman/multinli/
[6] Matched : https://www.kaggle.com/c/multinli-matched-open-evaluation ; Mismatched: https://www.kaggle.com/c/multinli-mismatched-open-evaluation

Model	Training Acc.	Test Acc.
(1) LSTM (Bowman et al., 2015)	84.4	77.6
(2) Classifier (Bowman et al., 2015)	99.7	78.2
(3) 300D tree-based CNN encoders (Mou et al., 2016)	83.3	82.1
(4) 300D SPINN-PI encoders (Bowman et al. 2016)	89.2	83.2
(5) 100D LSTMs w/ word-by-word attention (Rocktaschel et al., 2015)	85.3	83.5
(6) 300D mLSTM word-by-word attention model (Wang & Jiang, 2016)	92.0	86.1
(7) 200D decomposable attention model (Parikh et al., 2015)	89.5	86.3
(8) 200D decomposable attention model with intra-sentence attention (Parikh et al., 2015)	90.5	86.8
(9) Binary Tree-LSTM + Structured Attention & Composition + dual-attention (Zhao et al., 2016)	87.7	87.2
(10) 300D Full tree matching NTI-SLSTM-LSTM w/ global attention (Munkhdalai and Yu, 2016)	88.5	87.3
(11) 300D Syntactic Tree-LSTM (Chen et al., 2017a)	92.9	87.8
Human Performance (Gong et al., 2017)	97.2	87.7
Our model	90.3	87.4

Table 1. Performance on SNLI

Model	Test Acc.	
	Matched	Mismatched
(1)BiLSTM (Williams et al., 2018)	67.0	67.6
(2) Inner Att (Balazs et al., 2017)	72.1	72.1
(3) ESIM (Williams et al., 2018)	72.3	72.1
(4) Gated-Att BiLSTM (Chen et al., 2017b)	73.2	73.6
(5) Shortcut-Stacked encoder (Nie & Bansal, 2017)	74.6	73.6
(6) DIIN (Gong et al., 2017)	78.8	77.8
(7) Inner Att (ensemble) (Balazs et al., 2017)	72.2	72.8
(8) Gated-Att BiLSTM (ensemble) (Chen et al., 2017b)	74.9	74.9
(9) DIIN (ensemble) (Gong et al., 2017)	80.0	78.7
Human Performance (Gong et al., 2017)	88.5	89.2
Our model	75.1	74.7

Table 2. Performance on MultiNLI

3.2 Details of training

In order to initialize the words in the triplets, we used 300 dimensional Glove embedding (Pennington et al., 2014). For the relation vectors (the dimension is set to 20), we use a standard normal distribution to randomly initialize the values and then normalize each vector. Besides, for OOV words, we follow (Parikh et al., 2016) to initialize them by randomly selecting one from 100 random vectors. During the training, the word embedding including the 100 random vectors for OOV words are fixed while the embedding of the relations will be updated. The dimensions of the Triplet embedding Layer for relation and head/dependent-words are set to 20 and 300, respectively. And the dimensions of other hidden layers are set to 1,024. Besides, *Adadelta* method (Zeiler, 2012) is adopted for optimization, and the batch size is 32, the dropout ratio is 0.2, while the learning rate is 0.1.

3.3 Results and Analysis

Table 1 shows the results of different models on SNLI. The first group includes two baseline classifiers presented by Bowman et al. (2015). In model (1), they use LSTM to learn a representation for the passage, and then use the representation of the passage-pair to predict the judgment label. Model (2) uses a

traditional statistical classifier to predict the label with some handcrafted features such as the overlapped words, negation detection, etc.

In the second group, Model (3) and model (4) are two models based on passage encoding with tree structure. In model (3), Mou et al. (2016) considers tree-based CNN to capture passage-level semantics, while Bowman et al. (2016) use parsing and interpretation within a single tree-sequence hybrid model in model (4).

In the next group, models (5)-(11) are inter-passage attention-based models which are similar to ours. Model (5) and model (6) are LSTMs with word-by-word attention. In model (7), Parikh et al. (2016) decompose each passage into a set of words and then compare two word-sets of the passage-pair. They further improve the performance by adding intra-passage attentions in model (8). Our model is inspired by their approach. Specifically, in models (9)-(11): Zhao et al. (2016) adopt tree-LSTM and attention mechanism based on binary tree to generate semantic units and compare; Munkhdalai and Yu (2016) constructs a full n-ary tree to improve the performance, while Chen et al. (2017a) use a syntactic tree-LSTM to extract the inner dependency relations in the passage and compare between the passage-pair.

Table 1 shows that our model achieves an accuracy of 87.4%, which outperforms most of those models with tree structure (even those with complicated network architectures (Munkhdalai and Yu, 2016; Zhao et al., 2016)), and our model is more interpretable than the other models which are based tree structure. Specifically, our performance is better than Parikh et al. (2016) significantly though our model is inspired by theirs.

In the first group of Table 2, models (1)-(6) show some published best performances on MultiNLI. And models (7)-(9) are some ensemble models which have a very complicated network architecture. From this table, we can see that our performance is better than models (1)-(5) on matched set (even outperforms the ensemble models (7) and (8)), and better than models (1)-(5) on mismatched set. Please note that the structure of our model is more interpretable than that of others.

	Train Acc.	Test Acc.
Original model	90.3	87.4
(1) merged comparison	89.3	84.5
(2) replace self-attention with inter-attention	92.1	86.6
(3) remove cross-comparison	88.3	86.5

Table 3. Ablation study on SNLI set

3.4 Ablation analysis

Table 3 examines the effect of each major component. In order to check whether the strategy of performing individual comparison between *RHD* triplets works well in this task, model (1) directly compares one triplet with the merged representation from the triplet-set of the other side (in the Matching Layer). In this model, we compute one representation for each triplet by concatenating its three parts, align between these triplet representations of the sentence pair, and then compare each triplet with the merged representation from the triplet-set of the other sentence as in (Parikh et al., 2016). It shows that the proposed individual comparison strategy gives a better result.

In model (2), instead of using self-attention to generate the alignment weight between one triplet pair, we first yield a semantic vector representation for each triplet by concatenating their three parts and processing it with a linear transform. And then we use the multiplication of the vector representations of each triplet-pair as their alignment weight (as adopted in (Parikh et al., 2016)). When we do this alteration, the accuracy drops to 86.6% on the test set. This again proves that adopting individual comparison for each triplet-pair can describe their relatedness more accurately.

Last, we remove *cross-comparison* from our approach and list the result in model (3) to check the effect of this component. This component is mainly used to compare the words which are similar in semantics but play different syntactic/semantic roles. It shows that when we remove this component, the accuracy drops to 86.5%.

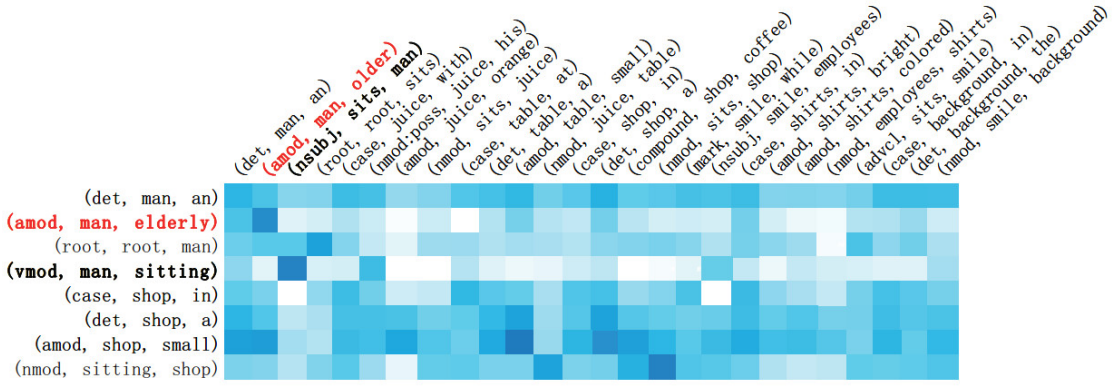

Figure 4. Triplet alignment weights for the triplet-pair in Figure 1. The darker color represents greater value. The triplets for *P* are on the top, and the triplets for *H* are on the left.

3.5 Visualization of Triplet Alignment

Figure 4 shows the alignment weights e_{ij} (i.e., the degree of relatedness between the triplets within the same sentence pair) of the example in Figure 1. In this figure, a darker color corresponds to a larger value of e_{ij}. From this figure, we can see that the related triplets do have larger alignment weights between them. For example, compared with other triplets of *H*, triplet (*amod, man, elderly*) in *H* (the red entry at the left of Figure 4) is more related to (*amod, man, older*) of *P* (the red entry at the top), which is echoed with a darker corresponding cell in Figure 4. Similarly, the corresponding cell for (*nsubj, sits, man*) of *P* and (*vmod, man, sitting*) of *H* shows a darker color, which also meets human judgment. This clearly shows that the alignment weights between these two triplet sets reflect the human interpretation closely.

4 Related Work

Early approaches for natural language inference usually adopted statistical models such as SVM (Joachims, 1998), CRF (Hatoriet et al., 2009) and so on, which employed hand-crafted features, and utilized various external resources and specialized sub-components such as *negation detection* (Lai and Hockenmaier, 2014; Levy et al., 2013). Besides, all the adopted datasets are very small.

After the SNLI corpus (Bowman et al., 2015) was released, a lot of work about natural language inference based on neural networks have been published in recent years (Bowman et al., 2016; Liu et al., 2016; Liu et al., 2016b; Munkhdalai and Yu, 2016; Mou et al., 2015; Sha et al., 2016). Basically, those neural network based approaches could be classified into 2 categories: (1) Merely computing the passage-embedding without introducing alignment (between the words in the sentences), and then comparing these passage-embedding to get the prediction (Bowman et al., 2016, Mou et al., 2016). (2) Decomposing the passage into some semantic units, comparing each semantic unit with the passage of the other side and then aggregating these comparisons (Parikh et al., 2016; Wang et al., 2017).

Within the first category, Bowman et al. (2015) used two LSTMs to get the representations of *P* and *H,* respectively. They then compare the vector representation of these two passages to predict the relationship between *P* and *H*. Besides, Bowman et al. (2016) used tree-LSTM to encode the representation of the passage. Although it uses the dependency relation in the passages to generate the representation, it cannot point out the difference among different words and is unable to catch the key information in the prediction step.

For the second category, Zhao et al. (2017) used the intermediate representations from tree-LSTM to compare *P* and *H*. It compares each node representation with the merged representation of the other passage, and generates the result bottom-up. In this way, it can extract the dependency relations in the passages and compare semantics in different granularities. However, they adopt the tree-LSTM to bottom-up generate the intermediate representations, and then directly compare each unit with the merged representation of the other passage. As the result, they cannot catch the key information and filter out irrelevant text.

Different from them, we transform *P* and *H* into two sets of *RHD* triplets, instead of comparing one *RHD* triplet of *P* with the merged information of the whole *H* (and vice versa), we directly compare a

RHD triplet of *P* with another *RHD* triplet of *H* to obtain the *individual comparison vector* (without merging). Specifically, when comparing two *RHD* triplets, each part in *RHD* triplet (i.e., *rel, head*, and *dep*) is compared separately and *cross-comparison* is adopted between nodes. Furthermore, we generate more precise alignment weights with these *individual comparison vectors* and self-attention mechanism. Consequently, our model is more interpretable than the previous models.

5 Conclusion

Inspired by how human judge the entailment relationship between the given *Premise (P)* and *Hypothesis (H)* text passages, we propose to transform the passages into two sets of word-pair dependency triplets, and then directly compare each triplet with every triplet from the other side to get the *individual comparison vector*. In this way, we can filter out the irrelevant information and judge the relationship between two passages more precisely. In order to further improve the comparison precision, we compare each part (i.e., "*rel*, *head*, and *dep*") of triplet separately and adopt *cross-comparison* to compare the related words which are with different syntactic/semantic roles. As these *individual comparison vectors* can describe the relatedness of the triplet-pair well, we use them and self-attention mechanism to generate the alignment weights between triplets from each passage. Afterwards, we use the alignment weights to incorporate these *individual comparison vectors* to yield the *one-side merged comparison vector* of one *RHD* triplet with the *RHD* triplet set of the whole other side. Finally, we aggregate those *one-side merged comparison vectors* to conduct the final overall entailment decision.

Acknowledgements

The research work descried in this paper has been supported by the National Key Research and Development Program of China under Grant No. 2017YFC0820700 and the Natural Science Foundation of China under Grant No. 61333018.

References

Jorge A Balazs, Edison Marrese-Taylor, Pablo Loyola, and Yutaka Matsuo. 2017. Refining Raw Sentence Representations for Textual Entailment Recognition via Attention. In *Proceedings of the 2nd Workshop on Evaluating Vector-Space Representations for NLP*, pages 51-55, Copenhagen, Denmark, September 7-11, 2017.

Pinaki Bhaskar, Somnath Banerjee, Partha Pakray, Samadrita Banerjee, Sivaji Bandyopadhyay and Alexander Gelbukh. 2013. A hybrid question answering system for Multiple Choice Question (MCQ). In: *Question Answering for Machine Reading Evaluation (QA4MRE) at CLEF 2013 Conference and Labs of the Evaluation Forum*, Valencia, Spain.

Johan Bos and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 628-635, Vancouver, October 2005.

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, Lisbon, Portugal, 17-21 September 2015.

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. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1466-1477, Berlin, Germany, August 7-12, 2016.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang and Diana Inkpen. 2017a. Enhanced LSTM for Natural Language Inference. In *Proceedings of the 55th Annual Meeting of the Association for Computional Linguistics*, pages 1657-1668 Vancouver, Canada.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Recurrent Neural Network-Based Sentence Encoder with Gated Attention for Natural Language Inference. In *Proceedings of the 2nd Workshop on Evaluating Vector-Space Representations for NLP*, pages 36-40, Copenhagen, Denmark, September 7-11, 2017.

Peter Clark and Qren Etzioni. 2016. My computer is an honor student-but how intelligent is it? Standard tests as measure of ai. *AI Magazine 37(1):5-12.*

Ido Dagan, Oren Glickman and Bernardo Magnini. 2005. The PASCAL Recognising Textual Entailment Challenge. In *International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment,* pages 177-190.

Qianlong Du, Chengqing Zong and Keh-Yih Su. Intergrating Structural Context and Local Context for Disambiguating Word Senses. The *Fifth Conference On Natural Language Processing and Chinese Computing & The Twenty Fourth International Conference On Computer Processing of Oriental Languages(NLPCC-ICCPOL 2016)*，Kunming, China, December 2-6, 2016, pages. 3-15

Dedre Gentner and Arthur B. Markman. 1997. Structure mapping in analogy and similarity. *American Psychologist, 52(1), 45-56*.

Dedre Gentner. 1983. Structure-mapping: A theoretical framework for analogy. *Congnitive science 7(2):155-170.*

Yichen Gong, Heng Luo and Jian Zhang. 2017. Natural Language Inference Over Interaction Space. CoRR abs/1709.04348. http://arxiv.org/abs/1709.04348.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artifical Intelligence and Statistics (AISTATS)* 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Sydney, Australia, 2006:905-912*

Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. 2007. Satisfying information needs with multi-document summaries. *Information Processing & Management* 43.6 (2007): 1619-1642.

Jun Hatori, Yusuke Miyao, and Jun'ichi Tsujii. On contribution of sense dependencies to word sense disambiguation. 2009. *Information and Media Technologies* 4.4 (2009): 1129-1155.

Michael Heilman, and Noah A. Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. 2010. *Human Language Technologies*: *The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.*

Thorsten Joachims. Making large-scale SVM learning practical. 1998. No. 1998, 28. *Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund.*

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR abs/1412.6980. http://arxiv.org/abs/1412.6980.*

Milen Kouylekov, and Bernardo Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment.*

Alice Lai, and Julia Hockenmaier. 2014. Illinois-LH: A Denotational and Distributional Approach to Semantics. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 329-334, Dublin, Ireland, August 23-24, 2014.

Omer Levy, Torsten Zesch, Ido Dagan and Iryna Gurevych. 2013. Recognizing partial textual entailment. In *Proceedings of the 51st Annual Meeting of the Asociation for Computational Linguistics*, pages 451-455.

Pengfei Liu, Xipeng Qiu, Jifan Chen and Xuanjing Huang. 2016a. Deep Fusion LSTMs for Text Semantic Matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1034-1043.

Yang Liu, Chengjie Sun, Lei Lin and Xiaolong Wang. 2016b. Learning Natural Language Inference using Bidirectional LSTM model and Inner-Attention. In *arXiv preprint arXiv*: 1605.09090.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learing Research*, 9(Nov):2579-2605.

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. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014),* pages 1-8, Dublin, Ireland, August 23-24 2014.

.Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan and Zhi Jin. 2016. Natural Language Inference by Tree-Based Convoluation and Heuristic Matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics,* pages 130-136, Berlin, Germany, August 7-12, 2016.

Tsendsuren Munkhdalai and Hong Yu. 2016. Neural tree indexers for text understanding. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers,* pages 11-21, Valencia, Spain, April 3-7, 2017.

Yixin Nie and Mohit Bansal. Shortcut-Stacked Sentence Encoder for Multi-Domain Inference. In *Proceedings of the 2nd Workshop on Evaluating Vector-Space Representation for NLP*, pages 41-45, Copenhagen, Denmark, September 7-11, 2017.

Rodney D. Nielsen, Wayne Ward, and James H. Martin. 2009. Recognizing entailment in intelligent tutoring systems. *Natural Language Engineering 15.4 (2009): 479-501.*

Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan and Alberto Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *Proceedings of EACL*, pages 409-416, Trento.

Sebastian Padó, et al. 2009. Measuring machine translation quality as semantic equivalence: A metric based on entailment features. *Machine Translation* 23.2-3 (2009): 181-193.

Ankur P. Parikh, Oscar Tackstrom, Dipanjan Das and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249-2255.

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 (EMNLP),* pages 1532-1543, October 25-29, 2014, Doha, Qatar.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský and Phil Blunsom. 2015. Reasoning about entailment with neural attention. In *arXiv-2015*. https://arxiv.org/abs/1509.06664

Lei Sha, Baobao Chang, Zhifang Sui and Sujian Li. 2016. Reading and Thinking: Reread LSTM Unit for Textual Entailment Recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2870-2879.

Nidhi Sharma, Richa Sharma and Kanad K. Biswas. 2015. Recognizing Textual Entailment using Dependency Analysis and Machine Learning. In *Proceedings of NAACL-HLT 2015 Student Research Workshop (SRW),* pages 147-153.

Hideki Shima, Hiroshi Kanayama, Cheng-Wei Lee, Chuan-Jie Lin, Teruko Mitamura, Yusuke Miyao, Shuming Shi and Koichi Takeda. 2011. Overview of NTCIR-9 RITE: Recognizing Inference in Text. In *Proceedings of NTCIR-9 Workshop Meeting*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutshever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural network from overfintting. The *Journal of Machine Learning Research*, 15(1):1929-1958.

Shuohang Wang and Jing Jiang. 2016. Learning Natural Language Inference with LSTM. In *Proceedings of NAACL-HLT 2016,* pages 1442-1451, San Diego, California, June 12-17, 2016.

Zhiguo Wang, Wael Hamza and Radu Florian. Bilateral Multi-Perspective Matching for Natural Language Sentences. *IJCAI(2017)*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of NAACL-HLT 2018*, pages 1112-1122, New Orleans, Louisiana, June 1-6, 2018.

Kai Zhao, Liang Huang and Mingbo Ma. Textual Entailment with Structured Attentions and Composition. In *Proceedings of Coling 2016, the 26th International Computational Linguistics: Technical Papers*, pages 2248-2258, Osaka, Japan, December 11-17 2016.

Jiang Zhao, Tian Tian Zhu and Man Lan. 2014. Ecnu: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 271-277, Dublin, Ireland, August 23-24, 2014.