Neural Simile Recognition with Cyclic Multitask Learning and Local Attention

Jiali Zeng,[1*] **Linfeng Song,**[2*] **Jinsong Su,**[1†] **Jun Xie,**[3] **Wei Song,**[4] **Jiebo Luo**[5]

[1]Xiamen University, Xiamen, China, [2]TencentAI Lab, Bellevue, USA
[3]Mobile Internet Group, Tencent Technology Co., Ltd, Beijing, China
[4]Capital Normal University, Beijing, China, [5]University of Rochester, Rochester NY, USA
lemon@stu.xmu.edu.cn, freesunshine0316@gmail.com, jssu@xmu.edu.cn,
stiffxie@tencent.com, wsong@cnu.edu.cn, jluo@cs.rochester.edu

Abstract

Simile recognition is to detect simile sentences and to extract simile components, i.e., *tenors* and *vehicles*. It involves two subtasks: *simile sentence classification* and *simile component extraction*. Recent work has shown that standard multitask learning is effective for Chinese simile recognition, but it is still uncertain whether the mutual effects between the subtasks have been well captured by simple parameter sharing. We propose a novel cyclic multitask learning framework for neural simile recognition, which stacks the subtasks and makes them into a loop by connecting the last to the first. It iteratively performs each subtask, taking the outputs of the previous subtask as additional inputs to the current one, so that the interdependence between the subtasks can be better explored. Extensive experiments show that our framework significantly outperforms the current state-of-the-art model and our carefully designed baselines, and the gains are still remarkable using BERT. Source Code of this paper are available on https://github.com/DeepLearnXMU/Cyclic.

Introduction

Simile is a special type of metaphor that compares two objects (called *tenor* and *vehicle*) of different categories using comparator words such as "*like*", "*as*" or "*than*". A Chinese simile sentence is shown in Figure 1, where the tenor "*Magnolia flower*" and the vehicle "*perfume*" are compared using comparator "*like*". Typically, simile recognition involves two subtasks (Liu et al. 2018): *Simile Sentence Classification*, which discriminates whether a sentence containing a comparator is a simile sentence, and *Simile Component Extraction*, which aims to extract the tenor and the vehicle in a simile sentence, respectively.

It is of great importance to study simile. Simile recognition is potentially beneficial for NLP applications, such as sentiment analysis (e.g. hate speech detection), dialogue understanding and question answering, because users sometimes use simile to express their emotions. In addition, simile recognition can help language learners to better under-
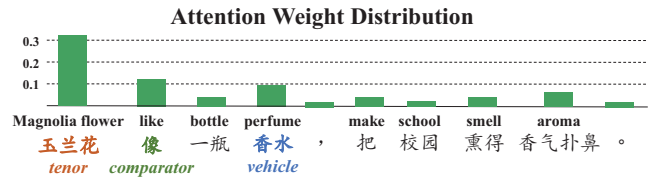

Figure 1: Attention weights generated by the simile sentence classifier of the conventional multitask learning (Liu et al. 2018), showing that the simile sentence classifier tends to focus more on simile components.

stand the implicit meanings expressed by simile in books and novels by highlighting simile components. However, simile recognition is very challenging, with one reason being that simile sentences have very similar syntactic and semantic structures to the normal sentences, hindering the feasibility of standard NLU techniques, such as syntactic and semantic parsing. Even though comparator words can provide some hints, they are also frequently used in literal comparisons, which introduces great ambiguity to this task.

Previous approaches of simile recognition are primarily based on handcrafted linguistic features and syntactic patterns (Li et al. 2008; Niculae and Yaneva 2013; Niculae 2013; Niculae and Danescu-Niculescu-Mizil 2014), which are inefficient on new languages and domains due to the extra time for feature engineering. Inspired by the successful applications of neural multitask learning on many NLP tasks (Liu, Qiu, and Huang 2016; Zhang and Weiss 2016; Luo et al. 2015; Miwa and Bansal 2016), Liu et al. (2018) investigated a standard multitask learning framework on simile recognition, which significantly outperforms the existing methods. Specifically, they apply a bi-directional LSTM (Hochreiter and Schmidhuber 1997) to encode the representations of each input sentence, and then the encoding results are taken as features shared by an attention-based sentence classifier, a CRF-based component extractor (Lafferty, McCallum, and Pereira 2001) and a language model that serves as an auxiliary task for additional supervision signals.

Despite their success, the multitask learning framework of Liu et al. (2018) suffers from two major drawbacks. **First**, simple parameter sharing is unable to fully exploit the se-

mantic interdependence between the two subtasks of simile recognition. It is intuitive that the results of these subtasks can be beneficial to each other. The potential simile components usually get higher attention weights during simile sentence classification. Taking Figure 1 as an example, the attention weights for tenor "*Magnolia flower*" and vehicle "*perfume*" are much higher than those of the other words. Therefore, simile component extractor can be more precise with the information about potential tenor and vehicle (attention distribution) from the simile sentence classifier. Moreover, it will be easier for simile sentence classification if we have identified the tenor and vehicle through component extraction, since they directly determine whether the sentence is a simile.

Second, both tenors and vehicles are usually close to comparators. In a standard simile benchmark (Liu et al. 2018), the average distances from tenors to comparators and from vehicles to comparators are 3.0 and 4.3, respectively, while the average sentence length is 29.5. As a result, the global attention mechanism used by Liu et al. (2018) can suffer from attention errors, since it considers all words in a sentence. Back to Figure 1, irrelevant words, such as the words after ",", distract the global attention model and own attention weights significantly larger than zero.

To overcome the above drawbacks, we propose a novel cyclic multitask learning framework with local attention for neural simile recognition. Figure 2 shows our framework, which captures the correlations of its subtasks by feeding the output of each subtask into the next. It organizes the subtasks as a cycle that is executed for K times, thus all subtasks further benefit from all others.

Taking $K = 1$ as an example, **first**, a Bi-LSTM encodes the input sentence and produces a sequence of word representations. **Then**, a local attention model is applied to the local sequence of word representations around the comparator word, and the induced local context vector is fed into a simile sentence classifier. **Next**, we concatenate the attention weights generated by the local attention and word representations, before sending the results to a CRF layer to extract simile components via sequence labeling. **Afterwards**, the label distribution and the word representations are concatenated as the input of the sentence decoder to reconstruct the original sentence. **Finally**, the decoder states are summed with the word representations, and the results are the input for the following simile classification.

Overall, our contributions are three folds:

- We propose a novel cyclic multitask learning framework for neural simile recognition. Comparing with standard multi-task learning, this framework better models the inter-correlation among its sub-tasks.

- We introduce a local attention mechanism for simile sentence classification. To our knowledge, no previous work has explored local attention on this task.

- Our framework shows superior performance over carefully designed baselines with or without pretrained BERT (Devlin et al. 2019), introducing the new state-of-the-art performance in the literature.
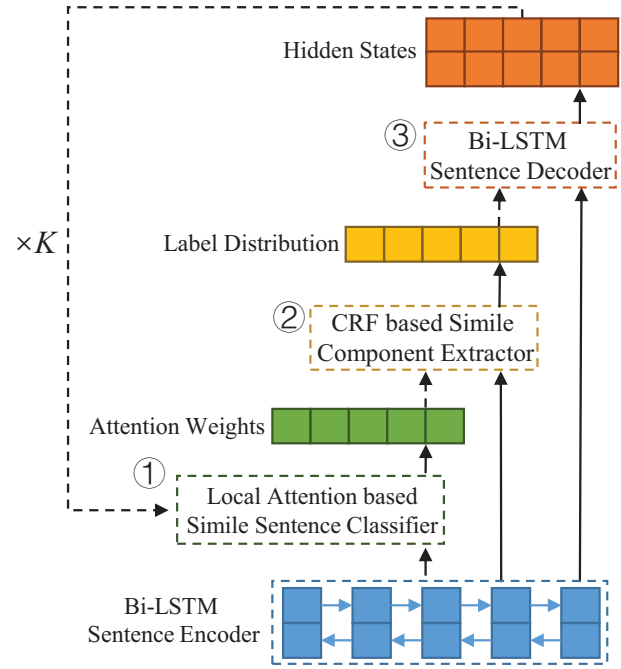

Figure 2: The architecture of our framework. Following previous work (Liu et al. 2018), ③ is taken as an auxiliary subtask for additional supervision.

Our Framework

In this section, we give a detailed description of our proposed framework. As shown in Figure 2, our cyclic framework concatenates a local attention based simile sentence classifier (①), a CRF based simile component extractor (②), and a Bi-LSTM sentence decoder (③) as a cycle, where the output of each module is fed as additional input to its successor module. In this way, the interdependence between different subtasks can be better exploited in our framework. In addition, it contains a Bi-LSTM sentence encoder that provides shared features to these modules (①, ② and ③). Note that our framework executes for K times, and the execution path is ①→②→③→① when $K = 1$.

Bi-LSTM based Sentence Encoder

Given an input sentence $X = (x_1, x_2, \ldots, x_N)$, we follow Liu et al. (2018) to first map its words to embeddings. Then, a Bi-LSTM is applied to produce a sequence of word representations $H = (h_1, h_2, \ldots, h_N)$ that are shared by our subtasks. The forward LSTM reads the sentence from left to right to learn the representation of each word x_i as $\overrightarrow{h}_i$. Similarly, the backward LSTM reversely scans the source sentence and learns each representation $\overleftarrow{h}_i$. Finally, for each word x_i, the representations from two LSTMs are concatenated to form the word representation $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$.

Local Attention based Simile Sentence Classifier

Simile sentence classification is a binary classification task determining whether a sentence contains any simile or not.

Global Attention Mechanism
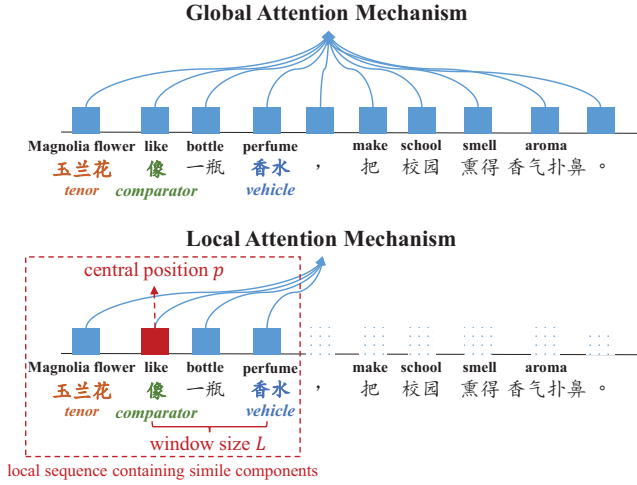

Local Attention Mechanism

Figure 3: Global attention mechanism and our proposed local attention mechanism.

As analyzed previously, the adopted global attention mechanism of Liu et al. (2018) considers all words. Hence, it is not suitable for the simile sentence classification, which mainly depends on the tenor and the vehicle around the comparator. To address this issue, we base our simile sentence classifier on a *local* attention mechanism. Compared with a global attention mechanism, our local attention mechanism only focuses on a dynamically choosed local context surrounding the comparator word.

We contrast our local attention mechanism with the global attention mechanism in Figure 3. For more details, since the comparator word, such as "like", "than" or "as", is given in each sentence, we first choose the position of the comparator word as the central position p, and then dynamically generate a context window size L as follows:

$$L = \frac{N}{2} \cdot \sigma(v_w^T tanh(\sum_{i=1}^{N} W_o h_i + W_p h_p)), \quad (1)$$

where h_p is the representation vector of the central word, W_p, W_o, and v_w are learnable parameters. As the next step, we perform an attention operation on the local sequence of word representations $\{h_{p-L}, ..., h_p, ..., h_{p+L}\}$ to generate a local context vector r via

$$r = \sum_{i=p-L}^{p+L} \lambda_i h_i, \quad (2)$$

$$\lambda_i = softmax(v_a^T tanh(W_o h_i)),$$

where W_o has been defined in Eq.(1) and v_a is a model parameter. Next, we stack a feed-forward network on r to induce high-level features. Finally, these features are used as inputs of a $softmax$ layer to conduct classification:

$$p(y|H) = softmax(W_c^0 ReLU(W_c^1 r)), \quad (3)$$

where W_c^0 and W_c^1 are model parameters[1].

---

[1] Right now we follow previous work to assume there being one or zero simile for each instance. For future multi-simile extension, we simply sum all rs before Equation 3.

As shown in Figure 2, there are two cases for simile sentence classification within our cyclic framework: for the first case, only word representations H are available for producing the local attention weights as additional input to simile component extractor, while for the other case, both H and the states $S=(s_1, s_2, ..., s_N)$ of the Bi-LSTM sentence decoder (③ in Figure 2) are available. We directly take the sum of H and S as H, i.e., $H = H + S$, to conduct simile classification. To make a unified definition, the loss function for simile sentence classification is defined as:

$$\mathcal{J}_{sc} = -log(p(y|H, S)). \quad (4)$$

Intuitively, S contains useful information of simile component extractor and sentence decoder, which can be directly propagated to simile sentence classifier by incorporating S into it.

CRF based Component Extractor

We follow Liu et al. (2018) to implement simile component extraction as a sequence labeling task. As mentioned before, the results of simile sentence classification can be beneficial to simile component extraction. Thus, we augment the word representations H with the attention weights generated from Equation 2:

$$\widetilde{H} = (\widetilde{h}_1, ..., \widetilde{h}_N),$$
$$\widetilde{h}_i = \begin{cases} [h_i, \lambda_i] & i \in \{p-L, ..., p+L\} \\ [h_i, 0] & else \end{cases}, \quad (5)$$

and then we stack a CRF layer on $\widetilde{H}$. Formally, the score of a predicted sequence $Y'=(y_1, y_2, ..., y_N)$, where $y_i \in D$ and $|D| = d$, is defined as

$$\varphi(\widetilde{H}, Y') = \sum_{i=1}^{N} A_{y_{i-1}, y_i} + \sum_{i=1}^{N} M_{i, y_i}, \quad (6)$$

where $A \in \mathbb{R}^{d \times d}$ is a transition matrix updated during training, and A_{y_{i-1}, y_i} records the transition score from label y_{i-1} to y_i; similarly, $M=(m_1, ..., m_N) \in \mathbb{R}^{N \times d}$ is the emission matrix and M_{i, y_i} indicates the score of assigning tag y_i to x_i. Specifically, m_i is a d-dimensional label distribution vector generated by feeding $\widetilde{h}_i$ into a single-layer network with activation function $tanh$ and a $softmax$ layer. Finally, the probability of the label sequence Y given the sentence X is

$$p(Y|\widetilde{H}) = \frac{exp(\varphi(\widetilde{H}, Y))}{\sum_{\tilde{Y}} exp(\varphi(\widetilde{H}, \tilde{Y}))}. \quad (7)$$

To train this extractor, we minimize the standard log-likelihood loss function:

$$\mathcal{J}_{ce} = -log(p(Y|\widetilde{H}) \\ = -\varphi(\widetilde{H}, Y) + log\sum_{\tilde{Y}} e^{\varphi(\widetilde{H}, \tilde{Y})}. \quad (8)$$

Bi-LSTM based Sentence Decoder

Due to the small number of available training instances, we follow Liu et al. (2018) to incorporate language modeling

into our cyclic framework as an auxiliary task, which can help Bi-LSTM encoder better model the sentence information.

For more details, we concatenate each label distribution vector m_i from component extractor with word representation h_i to produce the initial state s_0 of the sentence decoder:

$$\overline{h}_i = [h_i, m_i],$$
$$s_0 = W_d^0 \sum_{i=0}^{N} \overline{h}_i, \qquad (9)$$

where W_d^0 is a learnable matrix. Next, the forward LSTM takes the previous hidden state $\overrightarrow{s}_{t-1}$ and the previous word embedding $e(x_{t-1})$ as input to produce the hidden state s_t at the t-th timestep:

$$\overrightarrow{s}_t = \text{LSTM}(e(x_{t-1}), \overrightarrow{s}_{t-1}), \qquad (10)$$

and then predict the current word x_t in the following way:

$$p(x_t|x_{<t}) = softmax(\overrightarrow{W}_d^1 \overrightarrow{q}),$$
$$\overrightarrow{q} = tanh(\overrightarrow{W}_d^2 \overrightarrow{s}_t + \overrightarrow{W}_d^3 e(x_{t-1})), \qquad (11)$$

where $\overrightarrow{W}_d^*$, $* \in [1,2,3]$ are trainable matrix parameters. Formally, the loss function for the forward sentence decoder is defined as

$$\overrightarrow{\mathcal{J}}_{fd} = -\sum_{t=1}^{N} log(p(x_t|x_{<t})). \qquad (12)$$

Similarly, the backward decoder is the same as the forward decoder, but with different model parameters. Equations are omitted for space limitation. The backward loss function is defined as

$$\overleftarrow{\mathcal{J}}_{bd} = -\sum_{t=1}^{N} log(p(x_t|x_{>t})). \qquad (13)$$

Finally, the loss function for the whole decoder is defined as the sum of those in two directions:

$$\mathcal{J}_{sd} = \overrightarrow{\mathcal{J}}_{fd} + \overleftarrow{\mathcal{J}}_{bd}. \qquad (14)$$

Overall Training Objective

The final training objective over an instance, which contains a sentence, a simile tag, and a sequence of component labels, becomes

$$\mathcal{J} = \alpha \cdot \mathcal{J}_{sc} + \beta \cdot \mathcal{J}_{ce} + (1 - \alpha - \beta) \cdot \mathcal{J}_{sd}, \qquad (15)$$

where α, β (s.t. $\alpha + \beta < 1$) are non-negtive weights assigned beforehand to balance the importance among the three tasks.

Experiments

Settings

Data We evaluate our model on a standard Chinese simile recognition benchmark (Liu et al. 2018), where each instance contains *one* or *zero* similes. Table 1 shows the basic statistics of this dataset. We follow Liu et al. (2018) to conduct 5-fold cross validation: the dataset is first equally divided into 5 folds. For each time, 4 folds are used as training and validation sets (80% for training, 20% for validation), and the remaining fold is used for testing. For simile extraction, the components (tenors and vehicles) are tagged with the IOBES scheme (Ratinov and Roth 2009).

#Sentence	11,337
#Simile Sentence	5,088
#Literal Sentence	6,249
#Token	334K
#Tenor	5,183
#Vehicle	5,119
#Unique tenor concept	1,680
#Unique vehicle concept	1,972
#Tenor-vehicle pair	5,214

Table 1: Statistics of our simile dataset.

Hyper-parameters For fair comparisons, we use the same hyper-parameters as (Liu et al. 2018). In particular, we use their pretrained 50-dimensional Word2Vec (Mikolov et al. 2013) embeddings, which are updated during training. For efficient training, we only use the sentences with at most 120 words. The hidden sizes for Bi-LSTM encoder and decoder are 128. The parameters between sentence encoder and bi-directional sentence decoder are shared. Also, the word embeddings and the pre-softmax linear transformation in the sentence decoder are shared. The batch size is 80. The dropout rate is 0.5. We adopt *Adadelta* (Zeiler 2012) as the optimizer with a learning rate of 1.0 and *early stopping* (Prechelt 1998). The optimal hyper-parameters $\alpha=0.1$, $\beta=0.8$ are chosen using the validation set.

Contrast Models We compare the following baselines and models to study the effectiveness of our cyclic MTL:

- **ME (Li et al. 2008)**. It is a maximum entropy model taking tokens, POS and dependency relation tags as features.
- **MTL (Liu et al. 2018)**. A multitask learning framework, where the simile sentence classification, simile component extraction and sentence reconstruction are jointly modeled. It is the previous state-of-the-art system for simile sentence classification.
- **MTL-OP (Liu et al. 2018)**. An "Optimized Pipeline" introduced by Liu et al. (2018) for improving simile component extraction. It involves two steps: it first uses 1-best results produced by a model jointly training their simile sentence classifier and language model to filter simile sentences; then, another model jointly training simile component extractor and language model is used to extract simile components from these sentences. Note that this model is just a pipeline for decoding.
- **MTL**. Our implemantation of *MTL (Liu et al. 2018)*, where *local* attention is adopted instead of a global one.
- **MTL-Pip**. It is a degraded variant of our framework without cyclic connection. Note that this is also novel as no previous work has investigated it on this task.
- **MTL-Cyc**. Our cyclic multitask learning framework.

Effect of the executing number K

Table 2 shows the validation results of our *MTL-Cyc* framework regarding the executing time K, where we show the results of *MTL* when $K = 0$. There are large improvements for both simile classification (task ①) and simile component recognition (task ②) when increasing K from 0 to 1, show-

K	**F1-score for ①**	**F1-score for ②**
0	86.09	63.15
1	86.62	73.33
2	86.65	73.40
3	86.27	73.57
4	85.89	72.83

Table 2: Experimental results on the validation set with different execution number K, where standard MTL results are shown when $K = 0$.

Model	Precision	Recall	F1-score
ME (Li et al. 2008)	76.61	78.32	77.45
MTL (Liu et al. 2018)	80.84	92.20	86.15
①-Global (Liu et al. 2018)	77.51	88.95	82.84
①	79.76	88.25	83.79
MTL(①+②)	81.95	87.44	84.61
MTL(①+③)	81.45	88.96	85.04
MTL-Pip(①→②)	81.72	89.67	85.51
MTL-Pip(②→①)	81.50	89.26	85.20
MTL(①+②+③)	81.60	92.10	86.53
MTL-Pip(①→②→③)	81.39	**93.01**	86.81
MTL-Pip(②→①→③)	81.80	91.99	86.59
MTL-Cyc	**82.12**	92.60	**87.04***

Table 3: Main results on simile sentence classification. * indicates significant at $p < 0.01$ over *MTL(①+②+③)* with 1000 bootstrap tests (Efron and Tibshirani 1994; Koehn 2004). For the remaining of this paper, we use the same measure for statistical significance.

ing the effectiveness of stacking the subtasks into a loop. Further increasing K from 1 to 2 only results in marginal improvements for both subtasks while introducing more running time, and their performances slightly go down when enlarging K from 2 to 4. All the evidence above indicates that our cyclic framework converges quickly, making it more practically useful. Considering both efficiency and performance, we set $K = 1$ for all experiments thereafter.

Task 1: Simile Sentence Classification

Table 3 shows the experimental results on simile sentence classification. Overall, our *MTL-Cyc* exhibits the best performance, outperforming the previous state of the arts: *ME (Li et al. 2008)* and *MTL (Liu et al. 2018)* and our baselines. In addition, we have the following interesting observations:

Effect of Local Attention As shown in Table 3 (**Line 4-5**), when replacing the conventional global attention with our proposed local one, the performance of simile sentence classifier is improved by about 1 points. This confirms the hypothesis that focusing on the local context of comparator is more suitable for detecting simile sentences.

Effects of Simile Component Extraction and Sentence Reconstruction Here, we incrementally add simile component extraction and sentence reconstruction to explore their contributions to simile sentence classification under different frameworks: *MTL* and *MTL-Pip*. From Table 3 (**Line 6-13**), we draw some conclusions: **First**, when

jointly modeling two subtasks, *MTL(①+②)*, *MTL(①+③)*, *MTL-Pip(①→②)* and *MTL-Pip(②→①)* all significantly outperform the single task model ①. This indicates that there exists intense interdependence between the subtasks of simile recognition. **Second**, both *MTL-Pip(①→②)* and *MTL-Pip(②→①)* show much better performance than *MTL(①+②)*, demonstrating that our framework is able to better utilize the interdependence between subtasks than *MTL*. This is due to the utilization of bi-directional interaction between these two tasks: the previous task provides useful information to the subsequent task, meanwhile, the back propagation of the subsequent task can also positively affect its previous one. Furthermore, the performance of *MTL-Pip(①→②)* is better than *MTL-Pip(②→①)*. Thus, we believe that the direction of jointly modeling subtasks has an important effect on our framework. **Third**, jointly modeling all three subtasks (the last group in Table 3) is better than modeling two tasks (the second last group in Table 3), no matter which framework (*MTL-Pip* or *MTL*) is used. This result suggests that any subtask can provide useful information to other subtasks. **Finally**, the better performance of *MTL-Pip(①→②→③)* regarding *MTL-Pip(②→①→③)* confirms that it is better to stack ② upon ① (①→②) via pipeline. Probably, this is because simile sentence classification (①, binary classification) is generally easier than simile component extraction (②, sequence labeling), and thus it is more reasonable to finish the easy task before the difficult one.

Effect of Cyclic Multitask Learning Framework As shown in Table 3 (**Line 10-13**), we can see that our *MTL-Cyc* outperforms other contrast systems, including *MTL-Pip(①→②→③)*. Note that *MTL-Pip(①→②→③)* is a subset of our cyclic framework and has not been investigated before. *MTL-Cyc* is better than previous numbers and our strong baselines, demonstrating its effectiveness.

Task 2: Simile Component Extraction

Table 4 shows the comparison results on simile component extraction. Similar to the experimental results on Task 1, our *MTL-Cyc* and *MTL-Pip* still beats other models. Specially, our *MTL-Pip(①→②→③)* exhibits better performance than *MTL-OP (Liu et al. 2018)*, showing the effectiveness of information sharing between subtasks during training. Moreover, we discuss the results from the following aspects:

Effects of Simile Sentence Classification and Sentence Reconstruction We first investigate contributions of simile sentence classification and sentence reconstruction to simile component extraction via *MTL* and *MTL-Pip* frameworks. Here, we draw the following conclusions: **First**, as same as our results on Task 1, both *MTL-Pip(①→②)* and *MTL-Pip(②→①)* obtain better performance than *MTL(①+②)*. It confirms the superiority of our framework, which enables the involving subtasks to better benefit from each other than the conventional multitask learning. **Second**, *MTL(①+②)* shows worse results than *MTL(②+③)*. This may be owing to ③ (sentence reconstruction) bringing more supervision signals over the encoder than ① (binary classification). **Third**, *MTL-Pip(①→②)* significantly outperforms *MTL-Pip(②→①)*. This observation confirms

Model	Precision	Recall	F1-score
MTL (Liu et al. 2018)	55.99	69.89	62.11
MTL-OP (Liu et al. 2018)	61.60	73.61	67.07
②	54.98	66.47	60.18
MTL(①+②)	55.46	65.09	59.89
MTL(②+③)	58.07	69.53	63.29
MTL-Pip(①→②)	63.14	70.12	66.45
MTL-Pip(②→①)	56.87	67.75	61.84
MTL(①+②+③)	55.92	71.30	62.68
MTL-Pip(①→②→③)	**64.08**	71.60	67.63
MTL-Pip(②→①→③)	57.54	73.37	64.50
MTL-Cyc	63.16	**73.78**	**68.05***

Table 4: Main results on simile component extraction. * indicates significant at $p < 0.01$ over *MTL(①+②+③)*.
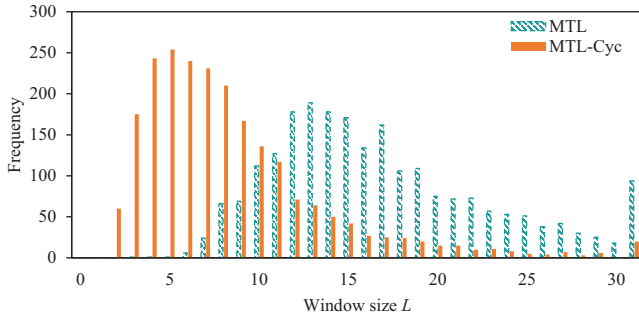

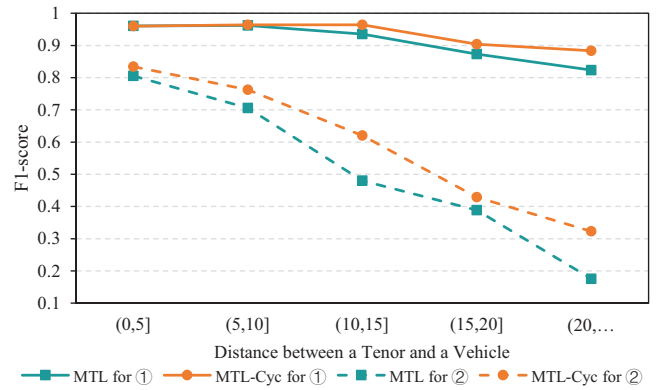Figure 4: Distribution of the predicted window size L.

Figure 5: F1-scores on different groups of test instances according to the distance between a tenor and a vehicle. Solid lines are results on simile sentence classification (①), dashed lines are results on simile component extraction (②).
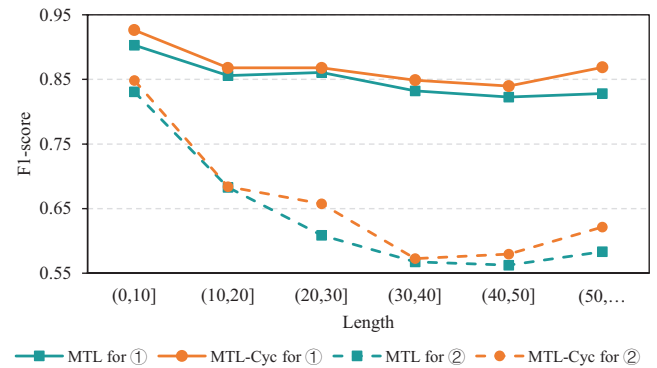

Figure 6: F1-scores on different groups of test instances according to sentence lengths.

that simile sentence classification is much easier than simile component extraction. Thus, the previous simile sentence classifier can provide useful information for the subsequent simile component extraction.

Effect of Cyclic Multitask Learning Framework   Our *MTL-Cyc* outperforms all *MTL-Pip* models, even with the same number of parameters. This result indicates that our cyclic setting mitigates the error propagation and truly benefits from leveraging the interdependence between subtasks.

Analysis

In order to better understand the individual effectiveness of our proposed local attention mechanism and cyclic multitask learning framework, we carry out additional analysis from the following aspects.

Distribution of Window Size   Figure 4 shows the distribution of the predicted window size L (Eq. 1) for our local attention on testset. The window sizes predicted by *MTL-Cyc* tend to be much smaller than those predicted by *MTL*. The underlying reason is that our *MTL-Cyc* allows its local attention module to be enhanced and better adjusted by the outputs and loss signals of the other tasks.

F1-scores over Distances between a Tenor and a Vehicle   We hypothesize that the difficulty of simile recognition increases as the distance between a tenor and a vehicle increases. To testify this, we display results on different groups of test examples regarding this distance in Figure 5. In both tasks, we find that the advantage of our framework

becomes more obvious as the distance increases. Particularly, the performance gap between our *MTL-Cyc* and *MTL* becomes greater up to 15 points for difficult instances.

F1-scores over Sentence Lengths   As shown in Figure 6, we compare our cyclic framework with standard MTL regarding different ranges of sentence lengths. Results show that our framework is consistently better than *MTL* in all groups, showing its robustness.

Effect using BERT (Devlin et al. 2019)   Recently, BERT has achieved great success in many NLP tasks by leveraging the rich knowledge within large-scale raw text via pretraining. To further demonstrate the effectiveness of our framework, we replace the pretrained word embeddings with the outputs of a Chinese BERT model[2], which is finetuned during simile training. Since the Bi-LSTM sentence decoder module (step ③ in Figure 2) works similarly with BERT by providing additional language modeling loss, we remove this module and directly construct a cycle with the other two tasks by feeding the outputs of simile component extraction (② in Figure 2) as additional inputs to simile sentence clas-

---

[2]https://github.com/ymcui/Chinese-BERT-wwm

Model(+BERT)	Precision	Recall	F1-score
Task 1: Simile Sentence Classification			
MTL(①+②)	83.31	93.61	88.16
MTL-Pip(①→②)	83.84	**94.63**	88.91
MTL-Cyc	**85.81**	94.43	**89.92**
Task 2: Simile Component Extraction			
MTL(①+②)	71.95	74.65	73.28
MTL-Pip(①→②)	72.17	**77.81**	74.88
MTL-Cyc	**73.97**	77.61	**75.74**

Table 5: Test results for simile recognition and extraction using pretrained BERT.

sification (① in Figure 2).

As shown in Table 5, our *MTL-Cyc* is still significantly better than *MTL* and *MTL-Pip* in both two subtasks given a strong pretrained BERT. Especially, the gains over *MTL* are almost 2.0 and 2.5 absolute points for simile sentence classification and simile component extraction, respectively. Both results confirm the superiority of our *MTL-Cyc* framework over standard MTL and other alternatives.

Related Work

Simile Recognition and its Applications   Similes have been studied in linguistics and psycholinguistics to explore how humans process similes, comparisons, metaphors, and the interplay among different components of these linguistic forms. Recently, simile recognition has wide applications in many tasks. Veale and Hao (2007) and Veale (2012) showed that the category-specific knowledge acquired from explicit similes can help to better understand figurative languages, such as metaphor and irony. Qadir et al. (2015) studied simile on sentiment classification, because people sometimes use simile to express their feelings instead of sentiment words. Since simile is very beneficial to other applications, simile recognition has received increasing interests in industrial and academic research. Li et al. (2008) introduced a maximum entropy model as simile sentence classifier and a CRF as simile component extractor. Niculae and Yaneva (2013) and Niculae (2013) recognized comparisons and similes through the use of syntactic patterns. Niculae and Danescu-Niculescu-Mizil (2014) distinguished simile in product reviews using a series of linguistic cues as features. Overall, these approaches were primarily based on handcrafted linguistic features and syntactic patterns. Inspired by successful applications of multitask learning, Liu et al. (2018) introduced a neural multitask learning framework. We are in line with Liu et al. (2018) in multitask modeling, but are different in that we consider the intercorrelation between different subtasks of simile recognition.

Multitask Learning   Recently, joint modeling multiple closely related tasks with shared representations has achieved great success on many NLP tasks, such as parsing and named entity recognition (NER) (Finkel and Manning 2010), NER and linking (Luo et al. 2015), text classification (Liu, Qiu, and Huang 2016), POS tagging and parsing (Zhang and Weiss 2016), extraction of entities and relations

(Miwa and Bansal 2016), event detection and summarization (Wang and Zhang 2017) and simile recognition (Liu et al. 2018). Unlike previous work, our framework further considers the interactions between subtasks by cyclic information propagation. On the simile recognition task, whose subtasks have strong intercorrelation between each other, our framework shows much stronger performance than conventional multitask learning framework.

Local Attention   In addition to neural machine translation (Luong, Pham, and Manning 2015; Yang et al. 2018), local attention mechanism has been shown effective on other NLP tasks, such as natural language inference (Sperber et al. 2018). We are the first to investigate the local attention mechanism on simile recognition.

Simile Recognition vs. Aspect-level Sentiment Analysis
The task of simile recognition looks similar to aspect-level sentiment classification (ASC) (Liu 2012; Pontiki et al. 2014). ASC is to determine the sentiment regarding a certain aspect, while simile recognition is to detect whether there is a simile regarding a comparator word in a sentence (Liu et al. 2018). However, simile recognition also requires extracting the corresponding *tenor* and *vehicle* if there is a simile sentence, while ASC does not require extracting any supporting evidence from text. Hence, the existing work for ASC can not be simply applied for simile recognition with a naive adaptation. More importantly, to the best of our knowledge, our framework is also novel and has large potential in the field of ASC. Specifically, the current state-of-the-art ASC model (Hu et al. 2019) shows that a decoding pipeline, which explicitly explores the intercorrelation among subtasks, surprisingly outperforms standard MTL (Li et al. 2019), even though decoding pipelines generally suffer from error propagation. This indicates that our cyclic-MTL may further improve ASC, as the joint training and cyclic flow of our framework can better model the intercorrelation among subtasks than a simple decoding pipeline. We leave studying our cyclic-MTL on ASC for future work.

Conclusion

We presented a novel cyclic multitask learning framework for simile recognition. Compared with conventional multitask learning, our framework can better model the dependencies among the subtasks. Extensive experiments and analysis strongly demonstrate the effectiveness of our framework.

In the future, we plan to investigate the generality of our framework on other multitask learning based NLP tasks. Besides, we will explore how to improve our framework by introducing variational networks, which have been widely used in many tasks (Zhang et al. 2016a; 2016b; Su et al. 2018a; 2018b).

Acknowledgments

References

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Efron, B., and Tibshirani, R. J. 1994. *An introduction to the bootstrap*. CRC press.

Finkel, J. R., and Manning, C. D. 2010. Hierarchical joint learning: Improving joint parsing and named entity recognition with non-jointly labeled data. In *ACL*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation*.

Hu, M.; Peng, Y.; Huang, Z.; Li, D.; and Lv, Y. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *ACL*.

Koehn, P. 2004. Statistical significance tests for machine translation evaluation. In *EMNLP*.

Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Li, B.; li Yu, L.; Shi, M.; and guang Qu, W. 2008. Computation of chinese simile with "xiang". *Chinese Information Processing*.

Li, X.; Bing, L.; Li, P.; and Lam, W. 2019. A unified model for opinion target extraction and target sentiment prediction. In *AAAI*.

Liu, L.; Hu, X.; Song, W.; Fu, R.; Liu, T.; and Hu, G. 2018. Neural multitask learning for simile recognition. In *EMNLP*.

Liu, P.; Qiu, X.; and Huang, X. 2016. Recurrent neural network for text classification with multi-task learning. In *IJCAI*.

Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Luo, G.; Huang, X.; Lin, C.; and Nie, Z. 2015. Joint entity recognition and disambiguation. In *EMNLP*.

Luong, T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Miwa, M., and Bansal, M. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*.

Niculae, V., and Danescu-Niculescu-Mizil, C. 2014. Brighter than gold: Figurative language in user generated comparisons. In *EMNLP*.

Niculae, V., and Yaneva, V. 2013. Computational considerations of comparisons and similes. In *ACL Student Research Workshop*.

Niculae, V. 2013. Comparison pattern matching and creative simile recognition. In *JSSP*.

Pontiki, M.; Galanis, D.; Pavlopoulos, J.; Papageorgiou, H.; Androutsopoulos, I.; and Manandhar, S. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval@COLING*.

Prechelt, L. 1998. Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks*.

Qadir, A.; Riloff, E.; and Walker, M. A. 2015. Learning to recognize affective polarity in similes. In *EMNLP*.

Ratinov, L., and Roth, D. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*.

Sperber, M.; Niehues, J.; Neubig, G.; Stüker, S.; and Waibel, A. 2018. Self-attentional acoustic models. In *Interspeech*.

Su, J.; Wu, S.; Xiong, D.; Lu, Y.; Han, X.; and Zhang, B. 2018a. Variational recurrent neural machine translation. In *AAAI*.

Su, J.; Wu, S.; Zhang, B.; Wu, C.; Qin, Y.; and Xiong, D. 2018b. A neural generative autoencoder for bilingual word embeddings. *Information Sciences*.

Veale, T., and Hao, Y. 2007. Learning to understand figurative language: from similes to metaphors to irony. *the Annual Meeting of the Cognitive Science Society*.

Veale, T. 2012. A computational exploration of creative similes. *Metaphor in Use: Context, culture, and communication*.

Wang, Z., and Zhang, Y. 2017. A neural model for joint event detection and summarization. In *IJCAI*.

Yang, B.; Tu, Z.; Wong, D. F.; Meng, F.; Chao, L. S.; and Zhang, T. 2018. Modeling localness for self-attention networks. In *EMNLP*.

Zeiler, M. D. 2012. ADADELTA: an adaptive learning rate method. *CoRR* abs/1212.5701.

Zhang, Y., and Weiss, D. 2016. Stack-propagation: Improved representation learning for syntax. In *ACL*.

Zhang, B.; Xiong, D.; Su, J.; Duan, H.; and Zhang, M. 2016a. Variational neural machine translation. In *EMNLP*.

Zhang, B.; Xiong, D.; Su, J.; Liu, Q.; Ji, R.; Duan, H.; and Zhang, M. 2016b. Variational neural discourse relation recognizer. In *EMNLP*.