Neural Multitask Learning for Simile Recognition

Lizhen Liu[†], **Xiao Hu**[†], **Wei Song**[†*], **Ruiji Fu**[‡], **Ting Liu**[§], **Guoping Hu**[‡]
[†]Information Engineering, Capital Normal University, Beijing, China
[‡]iFLYTEK Research, Beijing, China
[§]Harbin Institute of Technology, Harbin, China
{liz_liu7480,xiaohu,wsong}@cnu.edu.cn, {rjfu, gphu}@iflytek.com, tliu@ir.hit.edu.cn

Abstract

Simile is a special type of metaphor, where comparators such as *like* and *as* are used to compare two objects. Simile recognition is to recognize simile sentences and extract simile components, i.e., the *tenor* and the *vehicle*. This paper presents a study of simile recognition in Chinese. We construct an annotated corpus for this research, which consists of 11.3k sentences that contain a comparator. We propose a neural network framework for jointly optimizing three tasks: simile sentence classification, simile component extraction and language modeling. The experimental results show that the neural network based approaches can outperform all rule-based and feature-based baselines. Both simile sentence classification and simile component extraction can benefit from multitask learning. The former can be solved very well, while the latter is more difficult.

1 Introduction

A metaphor is a figure of speech that describes an object or action in a way that isn't literally true. Metaphors are common in human language. Shutova and Teufel (2010) reported that 241 among 760 sentences in an annotated corpus contain a metaphor. The use of metaphors helps to explain an idea or realize rhetorical effects through an analogical procedure. Metaphor analysis has been drawn more attention for expanding current natural language processing (NLP) to high-level semantic tasks (Carbonell, 1980).

Metaphors reflect creative thought of humans. On the other hand, inferring the meaning of a metaphor has to integrate background knowledge, which makes it difficult to automatically recognize metaphors in language. Previous work on metaphor recognition mainly depends on linguistic cues (Goatly, 2011) and selectional preference violation on a pair of concepts (Fass, 1991) or their domains (Mason, 2004). The domains can be created by knowledge bases such as WordNet (Mason, 2004) or based on automatic clustering (Shutova et al., 2010).

In this paper, we focus on a special type of metaphor—simile. A simile is a figure of speech that directly compares two things using connecting words such as *like*, *as*, *than* in English and "像" or "犹如" in Chinese. Due to the use of such comparators, it is much easier to locate similes compared with locating other types of metaphors. As a result, it is possible to collect and annotate large scale of simile sentences and investigate data driven simile recognition. This task is to find simile sentences and extract simile components, i.e., the *tenor* and the *vehicle*. The mined simile structures can potentially be used to support general metaphor analysis, where large scale training data is lacking.

However, simile recognition is still challenging due to the diversity of syntactic roles of a word and the distinction between metaphorical and literal comparisons. As shown in Table 1, a sentence containing a comparator may not trigger a simile. It is necessary to analyze the relationship between meanings of concepts. And It is also difficult to define a complete set of rules to extract the objects to be compared with high accuracy and coverage.

This paper presents an end-to-end neural network framework for simile sentence recognition. Specifically, we make following contributions:

- We build a dataset consisting of 11.3k sentences containing a frequently used comparator "像" for simile recognition in Chinese, which can support data-driven approaches. In contrast to English, datasets on simile or

*corresponding author

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1543–1553
Brussels, Belgium, October 31 - November 4, 2018.

1. 这个[孩子]$_{tenor}$ 壮得**像**一头[牛]$_{vehicle}$ This [boy]$_{tenor}$ is as strong *as* a [bull]$_{vehicle}$	Simile
2. 这个孩子长得**像**爸爸 The boy looks *like* his father	Literal
3. 他拍拍叔叔的肩膀,**像**是告诉他不要难过 He patted his uncle *as if* telling him not to be sad	Literal
4. **像**他这样的学生，应该更加努力 The students *like* him should work even harder	Literal

Table 1: Sentences that contain the comparator "像".

metaphor analysis are relatively less in Chinese. This dataset provides a new resource for related research. [1]

- We propose a neural multitask learning framework jointly optimizing three tasks: simile sentence classification, simile component extraction and language modeling. Simile classification is to determine whether a sentence with a comparator contains a simile, without knowing exactly what the tenor and the vehicle are. Simile component extraction aims to locate the tenor and the vehicle in a simile sentence. Intuitively, the two tasks should benefit each other. We design our model to enhance interactions between the two tasks. We also borrow the idea of Rei (2017) by incorporating a language modeling task, which attempts to predict neighbor words. All three tasks consider the whole sentence so that rich context information is involved.

- We conduct comprehensive experiments. The results demonstrate that the neural end-to-end framework is superior to feature-based and rule-based baselines and every single model can benefit from multitask learning. Simile sentence classification can be solved very well, while simile component extraction is more chellenging. With multitask learning enhanced classifier and extractor, a classification-then-extraction method achieves the best performance for simile component extraction.

2 Related Work

2.1 Metaphor/Simile Analysis

Metaphor analysis becomes active in recent years. The tasks include metaphor recognition, metaphor explanation and metaphor generation (Shutova et al., 2013; Veale, 1995; Jang et al., 2016).

Simile is a special type of metaphor with the comparator and it is relatively easier to locate metaphorical parts. Niculae and Danescu-Niculescu-Mizil (2014) aimed to distinguish a comparison from figurative or literal in product reviews using a series of linguistic cues as features. It is similar to simile sentence classification. So we take it as a baseline. In their work, they assumed that the components can be correctly recognized. In our work, we use an automated component extractor instead.

Syntactic patterns are often used for extracting potential simile components and semantic analysis is then used to distinguish similes from literal comparisons (Niculae and Yaneva, 2013; Niculae, 2013). The main limitation is that such pattern based method is difficult to deal with sentences with complex structures. As a result, the coverage is relatively small.

Qadir et al. (2016) used syntactic structures, dictionary definitions, statistical cooccurrence, and word embedding vectors to infer implicit properties in similes. Qadir et al. (2015) also built a classifier with lexical features, semantic features, and sentiment features to infer the affective polarity of simile in twitters. Veale and Hao (2007) and Veale (2012a) utilized knowledge generated by similes to deal with metaphor and irony, and Veale (2012b) built a lexical stereotype model from similes. These work demonstrates the wide applications of simile recognition.

In Chinese, Li et al. (2008) proposed a feature-based method for simile recognition. Their evaluation was done on a small dataset. The annotated data in this work is much larger.

2.2 Multitask Learning for NLP

Many researchers have proposed to jointly learn multiple tasks with shared representations (Collobert and Weston, 2008). Improvements are reported on joint models between closely related tasks, such as text classification (Liu et al., 2016), POS tagging and parsing (Zhang and Weiss, 2016), parsing and named entity recognition (NER) (Finkel and Manning, 2010), NER and linking (Luo et al., 2015), extraction of entities and relations (Miwa and Bansal, 2016). Bingel and Søgaard (2017) offered a systematic view of relations between different tasks.

[1]The dataset is at https://github.com/cnunlp/Chinese-Simile-Recognition-Dataset

3 Task and Data

3.1 Task Description

A metaphor is as a matter of cross-domain mappings in conceptual structure which are expressed in language. Lakoff and Johnson (2008) explains it as a mapping between the *target* and the *source*, corresponding to the terms *tenor* and *vehicle*. The tenor is the subject to which attributes are ascribed, while the vehicle is the object whose attributes are borrowed.

Simile can be seen as a special type of metaphor, which is signaled by explicit markers such as *like* or *as* in English and "像" or "犹如" in Chinese. We call such words **comparators** (Hanks, 2012). Notice that a sentence containing a comparator doesn't guarantee that it is a simile sentence. Consider the examples in Table 1. All sentences contain the comparator "像". The first two sentences form comparison structures, but the first one triggers a cross-domain concept mapping, while the second one is a literal comparison. The word "像" in the third sentence means *as if*, rather than forming a comparison, and the comparator in the fourth sentence is to give examples.

Simile Recognition task can be defined as: Given a sentence containing a comparator, determine whether it is a simile sentence, if so, extract the *tenor* and the *vehicle* from it.

Simile recognition involves two subtasks.

Simile Sentence Classification (SC). For a sentence containing a comparator, determine whether the comparator triggers a metaphorical comparison, in another word, whether the sentence is a simile sentence.

Simile Component Extraction (CE). For a simile sentence, extract text spans that are corresponding to the *tenor* and the *vehicle* objects respectively.

Both tasks have a realistic significance. Simile can be seen as a rhetorical device for making thoughts or expressions more vivid. Simile sentence classification could be used to provide a signal to evaluate rhetorical effects of writings. Simile component extraction is potentially useful for building cognitive knowledge base.

3.2 Data

We construct a dataset from Chinese student essays written in Mandarin Chinese. We focus on the comparator "像", which is the most often used simile comparator in Chinese. The data annotation involves the following steps: (1) we sampled sentences that contain the word "像" from a sentence index built on more than 20,000 student essays; (2) we asked two annotators to label every sentence as a simile sentence or not; (3) the annotators further annotated boundaries of simile components, the tenor and the vehicle, in simile sentences.

Two points are important for annotation: (1) the definition of simile and (2) the boundary of simile components.

We desire that a simile sentence should satisfy at least two standards. First, there exists explicit tenor and vehicle, which are from different semantic domains. Second, the tenor and vehicle should have similar properties. We provided a manual and positive/negative examples to annotators. Even so, there are still fuzzy cases. The two annotators first labeled a sample of 200 sentences independently and then we measured their agreement and asked them to discuss the disagreements. After discussion, they labeled another set of sampled sentences to check whether they really reached a consensus. This process iterated several times. Their final inner-annotator agreement on simile sentence classification can reach to 91%. Finally, they labeled all sentences in the whole dataset.

Next, the annotators should further label simile components in simile sentences, which are usually noun phrases. Boundaries of simile components are required to be annotated as compact as possible until they can't be simplified any more. In most cases, we asked the annotators to label the head noun phrase without a modifier. A modifier often plays a role as a *shared property*. For example, in the sentence 男孩的脸像一个红苹果(*The boy's face is like a red apple*), 脸(*face*) and 苹果(*apple*) would be annotated as the tenor and vehicle respectively, while 男孩的(*boy's*) and 红(*red*) are not included. In contrast, in the sentence 天气像孩子的脸说变就变(*The weather is like a child's face, which changes unpredictably*), 孩子的脸*(child's face)* is preferred to 脸(*face*) as a component, because 脸(*face*) alone can't capture the property well. We used one annotator's annotation of 200 sentences as the gold answer and the other's annotation as the prediction to computer the F_1 score, which is 93.57% on all components and 90.7% on tenors and 96.47% on vehicles.

Table 2 shows the basic statistics of our dataset.

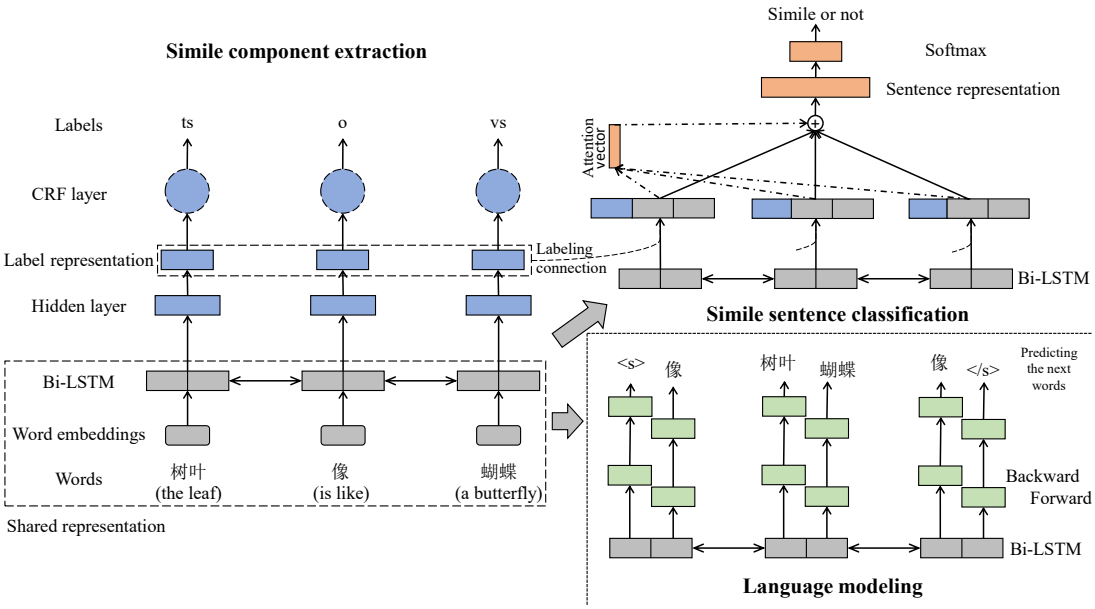

Figure 1: The proposed multitask learning framework, which jointly optimizes three tasks.

#Sentence	11337
#Simile sentence	5088
#Literal sentence	6249
#Token	334k
#Tenor	5183
#Vehicle	5119
#Unique tenor concept	1680
#Unique vehicle concept	1972
#Tenor-vehicle pair	5214
#Unique tenor-vehicle pair	4521
Avg. #token per tenor	1.033
Avg. #token per vehicle	1.056
Avg. #token per sentence	29.47
Avg. #pair per simile sentence	1.024

Table 2: Statistics of the annotated simile dataset

4 Multitask Learning Approach

4.1 Motivation

Intuitively, the two subtasks in simile recognition can benefit each other and the interactions between them should not be ignored. If the component extractor knows that a sentence contains a simile, it would be more confident to extract the tenor and the vehicle. On the other hand, if the component extractor tells the classifier that the tenor and the vehicle likely exist, the classifier gets additional information for decision.

Therefore, we propose a multitask learning approach to combine them. Our approach jointly optimizes three tasks: simile sentence classification, simile component extraction and language modeling. Language modeling is used as auxiliary task, which can help capture local information. Figure 1 illustrates the main framework, which is based on neural networks. We will first explain the representation layers that are shared by multiple tasks, and then introduce separate prediction layers for individual tasks.

4.2 Shared Representation

Word embedding layer. We first map words to dense distributed word embeddings. Since our dataset is not so large, we make use of pre-trained word embeddings, which are trained on a much larger corpus with Word2Vec toolkit (Mikolov et al., 2013).

Sentence representation layer. Recurrent neural networks (RNNs) have become the natural choice for handling sequential data to capture long-range dependencies. Given a sentence $X = (x_1, x_2, ..., x_n)$ containing n words as an input, the RNNs produce $H = (h_1, h_2, ..., h_n)$ as the hidden states to represent the semantic of partial sequence so far. Recently, Long Short Term Memory (LSTM) model (Hochreiter and Schmidhuber, 1997) has been proved more effective in various NLP tasks. Therefore, we use LSTM as the basic memory cell. At time step t, LSTM takes the hidden state from the previous time step and the word embedding from the current step as input, and produces a new hidden state, as shown in Formula 1.

$$\overrightarrow{h_t} = LSTM(x_t, LSTM(\overrightarrow{h_{t-1}})) \quad (1)$$

The LSTM architecture is sensitive to word or-

der, and the bidirectional LSTM (Schuster and Paliwal, 2002) allows model to look arbitrarily far at both the past and the future for the sake of grasping the whole sentence. Noted the forward LSTM as $\overrightarrow{h}$, and the backward as $\overleftarrow{h}$. Bidirectional LSTM concatenates the forward and backward states as the representation at the tth time step, i.e., $h_t = \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right]$.

4.3 Task 1: Simile Component Extraction

We view simile component extraction as a sequence labeling problem. We convert the annotated dataset to IOBES scheme (indicating *Inside, Outside, Beginning, Ending, Single*) (Ratinov and Roth, 2009). We use different prefixes to distinguish the tenor and the vehicle components. For example, *tb* and *vb* indicate the beginning of a *tenor* and a *vehicle* respectively.

4.3.1 Neural Sequence Labeling Model

Conditional Random Field (CRF) (Lafferty et al., 2001) is a standard solution in such scenario to exploit the dependency among labels.To further make use of the dense representation of words, we build a CRF layer on the shared representation layers following (Lample et al., 2016).

Formally, $H = (h_1, h_2, ..., h_n)$ is a sequence of hidden states produced by the bidirectional LSTM for a sentence X and $y = (y_1, y_2, ..., y_n)$ is the tag sequence, $y_i \in L$ and $|L| = k$. Define $\psi(H, y)$ as the score of the sequence.

$$\psi(H, y) = \sum_{t=0}^{n} A_{y_t, y_{t+1}} + \sum_{t=1}^{n} P_{t, y_t} \tag{2}$$

where $A \in \mathbb{R}^{k \times k}$ is a transition matrix and $A_{y_t, y_{t+1}}$ records the score of a transition from current label y_t to next label y_{t+1}; $P = (p_1, ..., p_n) \in \mathbb{R}^{n \times k}$ is the emission matrix and P_{t, y_t} represents the score of assigning tag y_t to x_t.

Here, h_t is the tth hidden state that is as assigned by the bidirectional LSTM. It is first mapped to a hidden layer through a feedforward layer. After a non-linear activation transition $\tanh$, the output of the hidden layer is mapped to a k-dimension vector p_t, through another feedforward layer.

$$p_t = W_p \cdot \tanh\left(W_t h_t\right) \tag{3}$$

where W_t and W_p are parameter matrixes. p_t can be seen as a tag score vector given the current word without considering context words.

Taking the whole state sequence into account, the probability of tag sequence y given sentence X is:

$$p(y|H) = \frac{\psi(H, y)}{\sum_{\widetilde{y} \in Y} e^{\psi(H, \widetilde{y})}} \tag{4}$$

where Y indicates all possible sequences. Learning algorithm attempts to optimize the model by maximizing the log-likelihood of correct tag sequence. Thus, the loss function is

$$\begin{aligned} E_{ce} &= -log(p(y|H)) \\ &= -\psi(H, y) + \log \sum_{\widetilde{y} \in Y} e^{\psi(H, \widetilde{y})} \end{aligned} \tag{5}$$

4.4 Task 2: Simile Sentence Classification

The second task is simile sentence classification. To fully exploit contextual information, we consider all words in a sentence. For each word, instead of using hidden state h_t only, we combine h_t and its score vector p_t as a representation s_t:

$$s_t = [h_t; p_t]$$

Since p_t is directly related to the component extraction task, this labeling connection operation increases the interaction between the two tasks.

However, words in a sentence should not contribute the same for classification. Intuitively, the words corresponding to the tenor or the vehicle or near comparators should be more important. Therefore, we introduce the attention mechanism, which was firstly proposed for neural machine translation (Bahdanau et al., 2014).

Given the sequence of expanded word representations $S = (s_1, s_2, ..., s_n)$, the attention vector is computed via:

$$\alpha = softmax\left(\tanh\left(W_\alpha S\right)\right) \tag{6}$$

where W_α is a parameter matrix. The semantic representation of the sentence is:

$$r = \alpha^T \cdot S \tag{7}$$

This representation is fed into an activation and a softmax layer to generate the probability distribution. The loss function is the negative log-likelihood of the correct classification tag:

$$E_{sc} = -\log\left(p\left(y|s\right)\right). \tag{8}$$

4.5 Task 3: Language Modeling

Although LSTM can capture long dependencies, simile structure may be more related to local contexts. In many cases, the comparator and the vehicle are near, and similes often have some collocations involving the comparator such as "像...一样(the same as)" or "就像(just like)". As a result, we attempt to emphasize such local information.

Inspired by (Rei, 2017), we also incorporate language modeling as an auxiliary task. For each word, we let the model predict the next word. In our case, the representation of each word h_t is firstly mapped into a low dimension vector space through a nonlinear transform.

$$\overrightarrow{m_t} = \tanh\left(\overrightarrow{W_m} \cdot \overrightarrow{h_t}\right) \tag{9}$$

And the vehicle word is predicted by maximizing the probability of the specific next word, which is generated by a softmax layer.

$$P\left(w_{t+1}|\overrightarrow{m_t}\right) = softmax\left(\overrightarrow{W_q} \cdot \overrightarrow{m_t}\right) \tag{10}$$

where, $\overrightarrow{m_t}$ indicates the forward language modeling specific features. W_m, W_q are trainable parameters.

The loss function for a sequence is defined as the sum of the negative log-likelihood of the predicted words.

$$\overrightarrow{E_{lm}} = -\sum_{t=1}^{n-1} \log\left(P\left(w_{t+1}|\overrightarrow{m_t}\right)\right) \tag{11}$$

We can also predict the previous words in the same way and get another loss function noted as $\overleftarrow{E_{lm}}$. The losses in double direction are summed to be the loss function for language modeling task.

$$E_{lm} = \overrightarrow{E_{lm}} + \overleftarrow{E_{lm}} \tag{12}$$

4.6 The Final Loss Function

The final loss function for each sentence is a weighted sum of task-specific loss functions.

$$E = \gamma \cdot E_{lm} + \delta \cdot E_{ce} + \epsilon \cdot E_{sc} \tag{13}$$

where γ, δ, ϵ are non-negative weights, which are used to control the importance of three tasks. In experiments, they are hyper-parameters assigned beforehand, and we constrain the sum of γ, δ and ϵ to one.

5 Evaluation

5.1 Settings

The dataset is randomly divided into 5 folds, 4 of which are used as training set and validation set (80% for training, 20% for validation), and the rest one fold is used as test set. All models were trained on the training set. The best hyper-parameters were gained based on the results on the validation set. The results reported were all evaluated on the test set.

We conduct word segmentation, part-of-speech (POS) tagging and dependency parsing with HIT-LTP[2].The word embeddings were pre-trained using Word2Vec (Mikolov et al., 2013), on a large essay corpus crawled from the web. We adopt the Theano framework (Theano Development Team, 2016) to implement neural network models.

The dimension of word embeddings is 50. The hidden size of LSTM is 128 for each direction. The dimension of activation layers for component extraction, simile sentence classification and language modeling are set to 64, 32 and 64 respectively. A dropout layer (Srivastava et al., 2014) is used between the word embedding layer and the bidirectional LSTM layer with the probability of 0.5. Moreover, early stopping (Prechelt, 1998) is adopted to finish the learning process. The AdaDelta (Zeiler, 2012) strategy is used for parameter optimization with a learning rate of 1.0.

5.2 Evaluating Simile Sentence Classification

5.2.1 Comparisons

We compare the following systems.

Feature based approaches. With manually designed features, we build two Random Forest classifiers to determine whether a sentence contains a simile. **Baseline1** follows (Niculae and Danescu-Niculescu-Mizil, 2014). It first extracts candidate simile components and then uses a classifier to determine whether they form a simile. We adopt our best neural component extractor (will be introduced in Section 5.3.1) to extract components. The classifier uses features including: (1) bag-of-words; (2) corresponding occurrence within constituents; (3) word embeddings of extracted components. Niculae and Danescu-Niculescu-Mizil (2014) also used domain specific information and lexicon knowledge, which our data lacks. **Baseline2** is based on (Li et al., 2008), which doesn't

[2] https://github.com/HIT-SCIR/pyltp

Model	Simile Classification		
	P	R	F_1
Baseline1	0.6523	0.4752	0.5498
Baseline2	0.7661	0.7832	0.7745
Singletask (SC)	0.7751	0.8895	0.8284
Multitask (SC+CE)	0.8056	0.8886	0.8450
Multitask (SC+LM)	0.8021	0.9105	0.8525
Multitask (SC+CE+LM)	**0.8084**	**0.9220**	**0.8615**

Table 3: Experimental results on simile sentence classification. SC: simile sentence classification; CE: component extraction; LM: language modeling.

need to identify components beforehand. The features include: (1) the tokens and POS tags of the words around the comparator within a fixed window (set to 5 in experiments); (2) the tokens, POS tags and dependency relation tags of the words that have dependency relations with the comparator.

Singletask(SC). This system is a simplified version of our proposed model in Section 4 by considering the simile sentence classification task only.

Multitask learning approaches. The full architecture is described in Section 4. To see their contributions, we add simile component extraction, language modeling and their combination incrementally.

5.2.2 Results

Table 3 shows the performance of the systems. The results are reported with the precision (P), recall (R), and their harmonic mean F_1 score (F_1).

The two feature based methods perform differently. Baseline1 performs poorly. The reason may be that the classification depends on the performance of component extraction, while even our best component extractor performs far from perfect, which brings error propagation. In addition, classifying with component related features only ignores much context, which further decreases the performance. Baseline2 considers context windows and outperforms baseline1 largely. This confirms our intuition that context information implies the semantic of simile expression.

Furthermore, we have other observations: (1) neural network based approaches largely outperform feature-based classifiers;(2) multitask learning approaches outperform every single task approach and other baselines. Both the component extraction and the language modeling task contribute for simile sentence classification. Component extraction improves the precision and language modeling improves both the precision and the recall. Combining them together can achieve the best performance. The improvement of F_1 score can reach to 3.3% compared with the best single task model.

5.3 Evaluating Simile Component Extraction

5.3.1 Comparisons

We compare the following systems for simile component extraction.

Rule based approach. We follow (Niculae and Yaneva, 2013) to design syntactic patterns for extraction. We convert the original patterns to fit the outputs of the parser we used.

CRF model. Since we view component extraction as a sequence labeling problem, a CRF model with manually designed feature templates is used as a baseline. Feature template is designed for every word. The features include the tokens and their POS tags within a fixed context window (set to 5 in experiments). We also use dependency parsing based features to capture dependencies between words.

Singletask(CE). We remove the simile classification and language modeling modules from the multitask learning framework introduced in Section 4 to build an end-to-end single task component extractor.

Pipeline approaches. Pipeline approaches first classify a sentence as simile or not, and then extract components from simile sentences. We investigate two combinations: RandomForest→CRF, we use baseline2 for sentence classification and CRF for component extraction; SingleSC→SingleCE, we use neural network based single task sentence classifier and component extractor.

Multitask learning approaches. We exploit simile sentence classification and language modeling modules to enhance component extraction in our multitask learning framework.

5.3.2 Results and Discussion

A component (i.e., the tenor or the vehicle) is judged to be correct only if both the boundary and the tag exactly match the gold answer. For a simile sentence, we should extract both the tenor and the vehicle rather than only partial components. Therefore, we use *pair-wise* level precision (P), recall (R) and F_1 score (F_1) for evaluation. A tenor-vehicle pair is viewed as correct only if both components are correct.

Table 4 shows the results of various systems and settings on two test sets. The first dataset consists

Model	Gold simile sentences			Whole test set		
	P	R	F_1	P	R	F_1
Rule based	0.4094	0.1805	0.2505		—	
CRF	0.5619	0.5907	0.5760	0.3157	0.3698	0.3406
Singletask (CE)	0.7297	0.7854	0.7564	0.5580	0.6489	0.5998
RandomForest → CRF		—		0.4591	0.4980	0.4778
SingleSC → SingleCE		—		0.5720	0.7074	0.6325
Multitask (CE+SC)		—		0.5409	0.6400	0.5861
Multitask (CE+LM)	0.7530	0.7876	0.7699	0.5741	0.7015	0.6306
Multitask (CE+SC+LM)		—		0.5599	0.6989	0.6211
Optimized pipeline		—		0.6160	0.7361	0.6707

Table 4: Experimental results on component extraction. Experiments on dataset of simile sentences assume that the sentence classifier is perfect. CE: component extraction; SC: simile sentence classification; LM: language modeling.

of all manually labeled simile sentences in the test set and the second dataset is the whole test set. We want to compare how component extraction systems work when they know whether a sentence contains a simile or not. We report and discuss the results from the following aspects.

The effect of simile sentence classification. First, we can compare the results in the middle column and the rightmost column in Table 4. It is clear that the component extraction systems work much better when they know whether a sentence contains a simile or not. Second, we can see that both pipelines (the feature-based and the neural network based) achieve a better performance compared with extracting components directly using either the CRF model or the neural single task model. Third, Multitask(CE+SC) doesn't bring significant improvements compared with the single task neural model. These observations indicate that simile sentence classification is suitable to be a pre-processing for simile component classification. It is necessary to further study how to use high level predictions (sentence classification) to learn better representations for consistently improving local predictions (simile component extraction).

Rule based, feature-based and neural models. We can see that even on gold simile sentences, the rule based method doesn't work well. The poor performance of the rule based approach is due to the following reasons. First, the rule-based method is difficult to deal with complex sentence structures. It often fails when there are multiple subordinate clauses. Second, the comparator "像" in Chinese has multiple syntactic roles, sometimes is used as a verb, sometimes is used as a preposition. Third, the accuracy of Chinese dependency parser still has room to be improved.

The CRF method performs significantly better, because it considers more contextual signals. Our neural single task model achieves large improvements on both datasets. This verifies the effectiveness of the end-to-end approach. Neural models can see a long range of context and learn features automatically. The word embeddings learned on external resources implicitly have semantic domain information, which is not only useful for generalization but also important for figurative language processing.

The effect of language modeling. Surprisingly, using language modeling as an auxiliary task is very useful, especially when dealing with noisy sentences. It gains a 1.3% F_1 improvement on the gold simile sentences due to the improvement on the precision and a 3% F_1 improvement on the whole test set due to a large improvement on the recall. Generally, language modeling may help learn better task specific representations, especially when data size is limited (Rei, 2017). Another reason may be that language modeling aims to make local predictions, the same as simile component extraction. Additional information from the same level may be more useful.

Some observations help understand the effect of language modeling. Figure 2 illustrates the relative distance of tenors and vehicles to the comparator. Both tenors and vehicles tend to occur near the comparator and have a clear preference on which side of the comparator. Tenors are more dispersed compared with vehicles, which may increase the difficulty. As shown in Table 5, the performance on identifying tenors is obviously worse than identifying vehicles in both settings.

Figure 3 shows the distribution of extracted components by two settings on the whole test set. We can see that with language modeling, Mul-

Model	Tenor			Vehicle		
	P	R	F_1	P	R	F_1
Singletask(CE)	0.7156	0.6935	0.7044	0.7789	0.8313	0.8043
Multitask (CE+LM)	0.6792	0.7881	0.7296	0.7393	0.9026	0.8128

Table 5: Experimental results on identifying individual types of components.

titask(CE+LM) makes predictions more aggressively compared with Singletask(CE) and tends to recognize more nearby components. We also observe that Multitask(CE+LM) identifies more correct UNK components, which are out of vocabulary or low frequency concepts. This means that language modeling leads the model to consider more local contextual patterns.

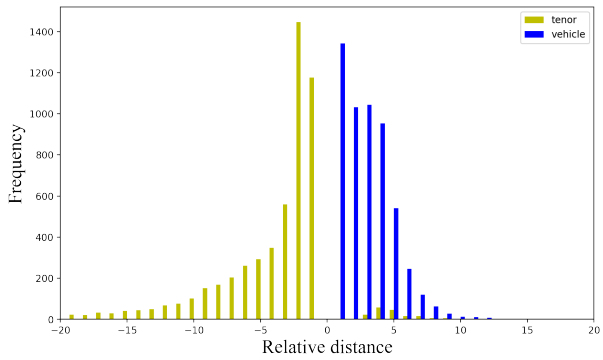

Figure 2: Relative distance of tenors and vehicles to the comparator in the dataset.

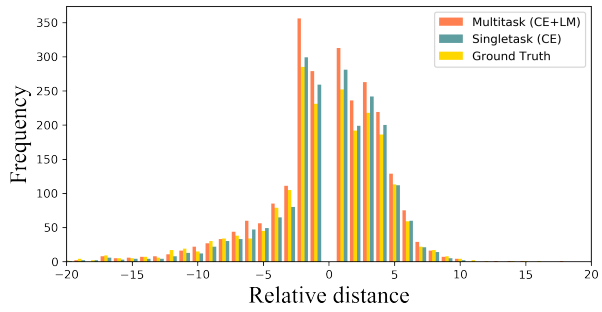

Figure 3: The distribution of extracted simile components on the whole test set.

Optimized Pipeline. According to the results and analysis, we can summarize that (1) simile sentence classification can achieve good performance by jointly optimizing three tasks; (2) simile components can be improved with language modeling as an auxiliary task; (3) simile sentence classification is suitable to be used as a pre-precessing for simile component classification. Therefore, we could build an optimized pipeline. We first use the enhanced simile sentence classifier to filter simile sentences and then use the enhanced component extractor to extract tenors and vehicles. As shown in Table 4, the optimized pipeline performs better than the strongest multitask learning setting.

However, in all settings, the precision scores are lower compared with the recall scores. This indicates that compared with identifying surface patterns, distinguishing metaphorical from literal meanings is much harder and more external knowledge should be incorporated.

6 Conclusion

This paper presented a study on simile recognition by exploiting neural networks. We construct a manually annotated dataset for advancing the research on simile analysis in Chinese. We propose a multitask learning framework, which jointly optimizes three tasks: simile sentence classification, simile component extraction and language modeling. The experimental results demonstrate the effectiveness of proposed approaches. It shows that simile sentence classification and simile component extraction both benefit from multitask learning. Simile sentence classification can achieve a high performance and simile component extraction still has a lot of room to improve.

In future, we plan to extend this work in several aspects: (1) enrich the simile component structure by adding shared properties or events so that the extracted structures would be more useful for metaphor processing; (2) improve representation learning for recognition by incorporating external knowledge; (3) apply simile recognition to study the use of figurative language in writings.

Acknowledgements

The research work is funded by the National Natural Science Foundation of China (No.61876113), High-level Teachers in Beijing Municipal Universities in the Period of 13th Five-year Plan (CIT&TCD20170322),Beijing Educational Committee Science and Technology Development Planned(No. KM201610028015),Humanity & Social Science general project of Ministry of Education(No.14YJC740087).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169. Association for Computational Linguistics.

Jaime G Carbonell. 1980. Metaphor: A key to extensible semantic analysis. In *Proceedings of the 18th annual meeting on Association for Computational Linguistics*, pages 17–21. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Dan Fass. 1991. met*: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 17(1):49–90.

Jenny Rose Finkel and Christopher D. Manning. 2010. Hierarchical joint learning: Improving joint parsing and named entity recognition with non-jointly labeled data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 720–728. Association for Computational Linguistics.

Andrew Goatly. 2011. *The language of metaphors*. Routledge.

Patrick Hanks. 2012. The roles and structure of comparisons, similes, and metaphors in natural language (an analogical system). *Prose (in honor of the Dickens Bicentennial)*, page 5.

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

Hyeju Jang, Yohan Jo, Qinlan Shen, Michael Miller, Seungwhan Moon, and Carolyn Rose. 2016. Metaphor detection with topic transition, emotion and cognition in context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 216–225. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, volume 951, pages 282–289.

George Lakoff and Mark Johnson. 2008. *Metaphors we live by*. University of Chicago press.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Bin Li, Li-li Yu, Min Shi, and Wei-guang Qu. 2008. Computation of chinese simile with "xiang". *Journal of Chinese Information Processing*, 22(6):27–32.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *IJCAI*.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888. Association for Computational Linguistics.

Zachary J Mason. 2004. Cormet: a computational, corpus-based conventional metaphor extraction system. *Computational linguistics*, 30(1):23–44.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116. Association for Computational Linguistics.

Vlad Niculae. 2013. Comparison pattern matching and creative simile recognition. In *Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora*, pages 110–114.

Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2014. Brighter than gold: Figurative language in user generated comparisons. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2008–2018. Association for Computational Linguistics.

Vlad Niculae and Victoria Yaneva. 2013. Computational considerations of comparisons and similes. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*, pages 89–95. Association for Computational Linguistics.

Lutz Prechelt. 1998. Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks*, 11(4):761–767.

Ashequl Qadir, Ellen Riloff, and Marilyn Walker. 2015. Learning to recognize affective polarity in similes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 190–200. Association for Computational Linguistics.

Ashequl Qadir, Ellen Riloff, and Marilyn A. Walker. 2016. Automatically inferring implicit properties in similes. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1223–1232. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Conll '09 design challenges and misconceptions in named entity recognition. In *CoNLL '09: Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2121–2130, Vancouver, Canada. Association for Computational Linguistics.

M. Schuster and K.K. Paliwal. 2002. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Ekaterina Shutova, Lin Sun, and Anna Korhonen. 2010. Metaphor identification using verb and noun clustering. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1002–1010, Beijing, China. Coling 2010 Organizing Committee.

Ekaterina Shutova and Simone Teufel. 2010. Metaphor corpus annotated for source - target domain mappings. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 3255–3261. European Languages Resources Association (ELRA).

Ekaterina Shutova, Simone Teufel, and Anna Korhonen. 2013. Statistical metaphor processing. *Computational Linguistics*, 39(2):301–353.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.

Tony Veale. 1995. Metaphor, memory and meaning: Symbolic and connectionist issues in metaphor interpretation.

Tony Veale. 2012a. A computational exploration of creative similes. *Metaphor in Use: Context, culture, and communication*, 38:329–343.

Tony Veale. 2012b. A context-sensitive, multi-faceted model of lexico-conceptual affect. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 75–79. Association for Computational Linguistics.

Tony Veale and Yanfen Hao. 2007. Learning to understand figurative language: from similes to metaphors to irony. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 29, pages 683–688.

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

Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1557–1566. Association for Computational Linguistics.