Exploring Accurate and Generic Simile Knowledge from Pre-trained Language Models

Shuhan Zhou[1], Longxuan Ma[2], and Yanqiu Shao[1](✉)

[1] School of Information Science, Beijing Language and Culture University, Beijing, China
yqshao163@163.com

[2] Research Center for Social Computing and Information Retrieval, Faculty of Computing, Harbin Institute of Technology, Harbin, China
lxma@ir.hit.edu.com

Abstract. A simile is an important linguistic phenomenon in daily communication and an important task in natural language processing (NLP). In recent years, pre-trained language models (PLMs) have achieved great success in NLP since they learn generic knowledge from a large corpus. However, PLMs still have hallucination problems that they could generate unrealistic or context-unrelated information. In this paper, we aim to explore more accurate simile knowledge from PLMs. To this end, we first fine-tune a single model to perform three main simile tasks (recognition, interpretation, and generation). In this way, the model gains a better understanding of the simile knowledge. However, this understanding may be limited by the distribution of the training data. To explore more generic simile knowledge from PLMs, we further add semantic dependency features in three tasks. The semantic dependency feature serves as a global signal and helps the model learn simile knowledge that can be applied to unseen domains. We test with seen and unseen domains after training. Automatic evaluations demonstrate that our method helps the PLMs to explore more accurate and generic simile knowledge for downstream tasks. Our method of exploring more accurate knowledge is not only useful for simile study but also useful for other NLP tasks leveraging knowledge from PLMs. Our code and data will be released on GitHub.

Keywords: NLP · Pre-trained language model · Simile knowledge

1 Introduction

A simile is a figure of speech that compares two things from different categories (called the tenor and the vehicle) via shared properties [17]. A tenor and a vehicle are usually connected with comparator words such as "like" or "as". For example, the sentence "The girl is as pretty as an angel." is a simile where the tenor is "The girl", the vehicle is "an angel", the comparator is "as ... as" and the shared property is “pretty”. Simile plays an important role in human language to make utterances more vivid, interesting, and graspable [26], comprehending similes is essential to appreciate the inner connection between different concepts and is useful for other natural language processing (NLP) tasks [8,20].

In recent years, pre-trained language models (PLMs) have achieved great success in NLP since they learn generic knowledge from a large corpus and could serve as a knowledge base [5,18]. Considerable attention has been paid to exploring simile knowledge from PLMs to solve downstream simile tasks, such as recognition, interpretation, and generation [4,8]. However, PLMs are known to suffer from hallucination problems [7,12,19], they could generate unrealistic or unfaithful information about the provided source content, which will impact their performance on downstream tasks. For example, when completing the blank in a simile sentence “Are you feeling ill? You are as __ as a ghost.”, a PLM may generate “creepy” instead of the expected shared property “pale”.

In this paper, we study how to explore more accurate and generic simile knowledge from PLMs. Specifically, we first train PLMs with three main simile tasks (recognition, interpretation, and generation). In this way, the PLMs can learn the shared semantic feature among different tasks and gain a better understanding of the simile knowledge. However, this understanding may be limited by the distribution of the training data. The performance of the model will drop when applied to unseen domains. To explore more generic simile knowledge, we further add semantic dependency features in the fine-tuning process. The semantic dependency feature serves as a global signal, helps the model learn simile knowledge shared among similar syntax structures, and enhances the model’s performance on unseen domains. During tests, we conduct experiments on both seen and unseen test sets to verify the effectiveness of our method. To sum up, our contributions are:

- We propose a novel method to explore more accurate and generic simile knowledge from PLMs.
- We test our model with both seen and unseen test sets. Experimental results demonstrate the effectiveness of our method and we give a detailed analysis of the results.
- Our code and data (including a new manually annotated simile data set) will be released on GitHub[1].

2 Related Work

In this section, we will introduce previous work related to this paper.

2.1 Simile and Metaphor

Metaphor is often used in human language to make speech more vivid and easy to understand [15]. [2] categorized metaphor into Noun phrases, Adjectives,

[1] https://github.com/realZsh/simile-tasks.

Table 1. Different metaphor categories. For similes, we use underline font to show **tenors** and use italic font to show *vehicles*.

Metaphor Category	Example	Is a simile?
Noun phrase	The judge is like *an angel.*	Yes
Adjective	The boy has a warm heart.	No
Verbal	He kills the seeds of peace.	No
Adverb-Verb	The child speaks France fluidly.	No
Verbal phrase	Raising little cats is like *taking care of children.*	Yes
Sentence	The man walks into the crowd like *a fish swims into the ocean.*	Yes

Verbs, and Multi-word. [10] defined metaphor as Nominal, Verbal (Subject-Verb-Object), Adjective-Noun, and Adverb-Verb. Table 1 shows examples of these categories. The Noun phrase metaphor is usually defined as a simile [4,8,10]. In this paper, we not only study the Noun phrase metaphor. Meanwhile, to test whether the trained model performs well on unseen domains, we construct a new test set. In this new test set, the tenor and vehicle can be verbal phrases/sentences that perform a similar role to Noun phrases. The examples of verbal phrases and sentences as simile components are shown in Table 1.

2.2 Tasks in Simile

The current simile study usually focus on recognition [1,11], interpretation [24], and generation [10]. The recognition task [10,14,22,25] is judging whether a triplet or a sentence contains a simile. The interpretation [11] assigns an appropriate interpretation to a simile expression [2] or infers the shared properties of the tenor and the vehicle [4,8,20]. The generation task generates a simile sentence [3,10,23,26] or the vehicle [4,20]. In this paper, we follow previous work and study the simile recognition/interpretation/generation (SR/SI/SG) tasks. Since there are not enough simile data that can be used for all three simile tasks. We construct the data we need based on existing SI data.

2.3 Exploring Simile Knowledge in PLMs

Previous simile work usually exploited the simile knowledge from PLMs for resolving downstream tasks. [20] fine-tune BERT [5] for simile recognition and simile component (tenor, shared property, and vehicle) extraction. [3] fine-tune BART [9] on the literal-simile pairs to generate novel similes given a literal sentence. [8] design a simile property probing task to let the PLMs infer the shared properties of similes for the interpretation task. [4] propose an Adjective-Noun mask Training method to explore simile knowledge from BERT for simile interpretation and generation tasks. [10] fine-tune a GPT-2 [18] model for simile generation. In this paper, we also study how to explore simile knowledge from PLMs. However, different from previous work, we investigate how to leverage three simile tasks to explore more generic simile knowledge from PLMs.

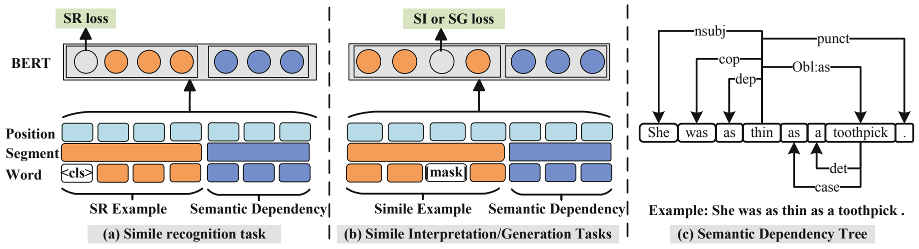
SR loss
BERT
Position
Segment
Word
<cls>
SR Example
Semantic Dependency
(a) Simile recognition task
SI or SG loss
[mask]
Simile Example
Semantic Dependency
(b) Simile Interpretation/Generation Tasks
nsubj
cop
dep
punct
Obl:as
She was as thin as a toothpick .
det
case
Example: She was as thin as a toothpick .
(c) Semantic Dependency Tree

Fig. 1. Demonstration of the training method and semantic dependency.

3 Our Proposed Method

In this section, we formalize the simile recognition/simile interpretation/simile generation (SR/SI/SG) tasks and introduce our method in detail. For a fair comparison with previous work [4,8], we use BERT-base [5] as the backbone of our model. Figure 1 shows the model structure of SR/SI/SG tasks.

3.1 Training of Simile Recognition (SR) Task

We follow previous work [10,11] and define SR as a binary classification task. The SR model needs to distinguish whether an input sequence contains a simile. The input to the SR model is a sequence and the output is a binary label: True for simile and False for literal. The only common feature between simile data and literal data is that they both contains the comparator words [11]. For example, the sentence "the boy runs like a deer." is a simile, but the sentence "the girl looks like her mother." is literal.

Following the original BERT paper, we use the first output position (a special token <cls>) to calculate the classification score, such as (a) part in Fig. 1. We denote the corresponding output vector of <cls> as E_{cls}. Then the final score $\mathcal{S}$ of the input sequence is calculated as follows:

$$\mathcal{S} = \sigma(W_2 \cdot \mu(W_1 \cdot E_{cls} + b_1) + b_2), \tag{1}$$

where $W_{1,2}$ and $b_{1,2}$ are training parameters; σ/μ is the sigmoid/tanh function, respectively. The example with $\mathcal{S} \geq 0.5$ is classified as a simile, otherwise literal. The training loss is cross-entropy between predicted labels y_i and ground-truth label $\bar{y}_i$:

$$\mathcal{L}_{SR} = -\frac{1}{N}\sum_{i=1}^{N}(\bar{y}_i logP(y_i)) \tag{2}$$

where N is the number of training examples. After this fine-tuning, we can test the model on the SR test sets. We input an example and verify whether the SR model gives a correct classification for it.

Table 2. Examples for simile interpretation/generation tasks. We place the correct answer in the first position in these examples. In real data, the position of the correct answer is randomly placed. During training, the model learns to recover the [MASK] word. During the test, the model needs to select one answer from the 4 candidates.

Task	Example	Candidates
SI	My client is as [MASK] as a newborn lamb.	**A**. innocent. **B**. delicious. **C**. legal. **D**. guilty.
SG	The participant swims like a [MASK].	**A**. dolphin. **B**. plait. **C**. depiction. **D**. pod.

3.2 Training of Simile Interpretation (SI) and Simile Generation (SG) Tasks

Following the previous simile interpretation (SI) and simile generation (SG) work [8,20], we define the training of SI and SG as a masked language model task where the BERT learns to recover the masked words, such as (b) part in Fig. 1. Two examples are shown in Table 2. In SI, the masked word is the shared property. In SG, the masked word is the vehicle.

During the test, we also follow the previous work [8,20] and define SI/SG as a multi-choice task which chooses an answer from 4 candidates. Given an input simile sentence or dialogue with a masked shared property/vehicle, the SI/SG model needs to select the correct property/vehicle from the candidates, respectively. We use the masked-word-prediction heads of BERT to compute the probability for each candidate. The candidate with the highest probability will be chosen as the final choice.

3.3 Training with Semantic Dependency Features

Through the training process with SR/SI/SG, the PLM learns to use simile knowledge for three different simile tasks. However, the distribution of the training data may restrict the model's performance when applied to unseen domains. To this end, we enhance the PLM with global semantic dependency information, which can help the model learn simile knowledge across different syntax structures. This more generic simile knowledge can help the model's performance on unseen domains.

We adopt the semantic dependency tool[2] to get the semantic dependency tree of each input sequence. One example is shown in (c) part of Fig. 1. The dependency tree for "She was as thin as a toothpick." is a list of tuples: "[('ROOT', '.', 'thin'), ('nsubj', 'thin', 'She'), ('cop', 'thin', 'was'), ('dep', 'thin', 'as'), ('case', 'toothpick', 'as'), ('det', 'toothpick', 'a'), ('obl', 'thin', 'toothpick'), ('punct', 'thin', '.')]". The word "thin" is the root of this tree and please refer to [13] for the definition of each semantic dependency relation.

[2] https://stanfordnlp.github.io/CoreNLP.

Table 3. Statistics of datasets.

Dataset	Train/Dev/Test	Words/Example	Data Format
MSP-original (for SI)	4,510/-/1,633	12.2	sentence
MSP-modified for SG	4,510/-/1,633	12.3	sentence
MSP-modified for SR	7,216/902/902	12.3	sentence
New test set	-/-/957	30.6	three-turn dialogue

For the SR task, we can directly use the semantic dependency results. However, in SI or SG task, key simile component such as the vehicle "toothpick" of the above example is masked. We change the example to "She was as thin as a UNK.", where UNK represents the [MASK] vehicle. Then the output semantic dependency tree changes to "[('ROOT', '.', 'thin'), ('nsubj', 'thin', 'She'), ('cop', 'thin', 'was'), ('dep', 'thin', 'as'), ('case', 'UNK', 'as'), ('det', 'UNK', 'a'), ('obl', 'thin', 'UNK'), ('punct', 'thin', '.')]". In this way, the model is aware of the semantic dependency tree of the input sentence but does not see the masked word.

The final input to BERT is the concatenation of the semantic dependency tree and the original sentence. We use different segment embedding to distinguish the data example and its semantic dependency information, such as the (a)/(b) part of Fig. 1.

After training, we test with two different settings, one is the MSP test set, and the other is an unseen test set that is newly constructed by us. Next, we will introduce the data sets.

4 Experimental Setup

4.1 Datasets

We use simile data sets with "as ... as" comparator since the shared property naturally exists in the comparator, which is suitable for our experiments since we want conduct all SR/SI/SG tasks with this data. This kind of simile data can be used for all three simile tasks. The data statistics are shown in Table 3 and we introduce the data details next.

MSP Dataset (for SI Task). Since we could not find enough data for all three simile tasks, we construct the required data based on a recently released simile benchmark. The multi-choice simile probe (MSP) data [8] is originally proposed for SI task. It has a total of 5,410 training examples and 1,633 test examples. All examples in MSP are simile sentences with comparator "as ... as". Each example in the MSP test set has three distractors for the shared property. During training, the model learns to recover the masked property in MSP training data. During the test, the model needs to choose the correct answer from 4 candidates in the MSP test set.

Table 4. Relations in ConceptNet we used to find distractors. "<->" means Symmetric relation for A and B. "->" means Asymmetric relation that A entails B.

Relation:	*Definition*
RelatedTo:	*The most general relation. There is some positive relationship between A and B, but ConceptNet can't determine what that relationship is based on the data. Symmetric. exercise <-> fit*
IsA:	*A is a subtype or a specific instance of B; every A is a B. This can include specific instances; the distinction between subtypes and instances is often blurry in language. This is the hyponym relation in WordNet. car -> vehicle; Mexico -> Country*
Causes:	*A and B are events, and it is typical for A to cause B. run -> tired*
Desires:	*A is a conscious entity that typically wants B. Many assertions of this type use the appropriate language's word for "person" as A. person -> respect*
DistinctFrom:	*A and B are distinct member of a set; something that is A is not B. Symmetric. red <-> blue; June <-> May*
SymbolOf:	*A symbolically represents B. blue -> cold*
MannerOf:	*A is a specific way to do B. Similar to "IsA", but for verbs. auction -> sale*
LocatedNear:	*A and B are typically found near each other. Symmetric. computer <-> table*
CausesDesire:	*A makes someone want B. hungry -> eat food*
MadeOf:	*A is made of B. porcelain -> ceramic*

MSP-Modified Data (for SG Task). To perform the SG task, we introduce a modified version of MSP. During training, we mask the vehicle and train the model to recover it. During the test, we provide 4 vehicle candidates for the multi-choice task. Besides the real vehicle, the other 3 distractors are constructed with ConceptNet [21]. The ConceptNet is a knowledge graph that connects words and phrases of natural language with labeled relations [21]. We show 10 relations of ConceptNet in Table 4. They are used to find the related concepts to the vehicle as the distractors. For the example "She was as thin as a toothpick.", the vehicle is the word "toothpick". We find that "toothpick" is usually located near to (LocatedNear) "food" and can be made of (MadeOf) "plastic" or "wooden". So the three distractors can be "food, plastic, wooden". When we find more than three distractors with the relations in Table 4, we randomly choose 3 of them as the final distractors. Notice that there are a few cases we could not find enough distractors, we manually construct distractors for these cases.

MSP-Modified Data (for SR Task). Similarly to the SG task, we introduce another modified version of MSP for the SR task. Since the SR task needs both simile examples and literal examples [10,11], we use certain relations in ConceptNet to obtain the literal data we need. For example, we replace the tenor "his muscle" in the simile example "his muscle is as hard as a rock" with the phrase "a stone", the Synonym concept of "a rock", then we get a literal sentence "a stone is as hard as a rock". This is different from replacing "his muscle" with a random word such as "air". Because the sentence "air is as hard as a rock" does not have a practical meaning. If we use "air is as hard as a rock" as a literal sample to train an SR model. The model may classify this sample as literal by identifying that it is against common sense. Instead, when we use the literal sentence "a stone is as hard as a rock", the SR model needs to use simile knowledge to judge whether this example is a simile. The knowledge is that simile only exists when comparing things from different categories. "stone" and "rock" are in the same category so this sentence is literal. Besides the Synonym relation, we can also use other relations of the vehicle including DistinctFrom/IsA/RelatedTo/SimilarTo in ConceptNet to find a concept to replace the tenor. When we find more than one distractor, we randomly choose one of them as the literal sentence. By this method, we not only obtain the required training literal data but also has more difficult literal data. Because the syntax structure of the literal data is the same as the original simile example but the semantic information is different. These literal examples will help the model to learn more accurate simile knowledge. Finally, we obtain 9020 examples. We randomly split this data into train/dev/test (8:1:1) to train our model. During training, the model learn to give a higher/lower score for the simile/literal data. During the test, the model assigns a score for the input. In both training and testing, an example with a score ≥ 0.5 will be set as simile, <0.5 will be set as literal.

A New Test Data (for SR/SI/SG Task). After the above data set construction, we now have the training/testing MSP sets for SR/SI/SG tasks. We denote the MSP test sets as a seen set because the training and testing data are in a similar domain and similar range of length. To test whether our method can help to explore more generic simile knowledge, we provide unseen test sets for SR/SI/SG tasks.

The new test data is collected from Reddit-dialogue corpus [6] which has ∼15 million English dialogues. The dialogues are comments from the Reddit forum and each dialogue has three turns. We extract 1,000 dialogue examples from the Reddit dataset with three rules. First, the dialogue length is around 30 tokens so it is informative and not too long. Second, the last turn must contain a comparator "as ... as" with an adjective word in the comparator. Third, we use the semantic dependency tool to ensure that the tenor and vehicle are in the response. Then we manually annotate whether they are similes or literal. For the simile sentences, we further check whether the tenor and vehicle labeled by the semantic dependency tool are correct. Notice that we do not make any change to the data. Therefore, for dialogue examples that tenor or vehicle is missing, we withdraw this example even it contains a simile. We make sure that all simile components are in the example so that we can use it for all simile tasks. We finally have 486 simile examples and 471 literal examples, total 957 examples. When testing on SI/SG, we construct the distractors using the same method as we construct MSP-modified data. For the examples in this new test set that we could not find enough vehicle distractors, we randomly choose the vehicles from other dialogues as the distractors.

The new test set is different from the training data (MSP) in the following respects: 1) the data format is dialogue and the length is much longer than data in MSP; 2) the tenor and vehicle in dialogue can be verbal phrase or sentence, which is different from the noun phrase in MSP. We use the new test set to verify whether our method can perform well on a different simile distribution compared to MSP.

4.2 Baselines

We introduce the baselines we used in this section.

Baselines for SR. BERT-base is fine-tuned on the MSP modified SR training set. The checkpoint for test is selected based on the performance on the corresponding dev set.

Baselines for SI/SG. The first baseline is a BERT-base model without fine-tuning with the data sets in this paper. It takes the input with key simile component masked and predicts the masked words. The second baseline is BERT-ANT [4] which is trained with masked word prediction with a number of metaphor data. It is based on a BERT-large-uncased model and can solve the SI and SG tasks in a unified framework of simile triple completion. For example, when giving tenor = fireman and vehicle = bull, BERT-ANT can generate a list of words including the shared property like “strong” or “brave”. When performing our SI/SG tasks, we match the candidates of each example with the output list of BERT-ANT. An example is counted correct if the ground truth answer is listed before the other three distractors. The BERT-Probe baseline is from [8] that fine-tuned BERT with MSP-original data for simile interpretation task. To compare both SI and SG tasks with this baseline, we further fine-tuned the BERT-Probe model with MSP-modified SG training data and report its results on the MSP-modified SG test data.

Our Models. Besides the fully fine-tuned model, we also provide several settings for our model. (- SR training) means we remove the simile recognition data in the unified training process. Similarly, (- SI training) and (- SG training) means we remove the SI and SG data in training, respectively. (- Semantic Dependency) means we do not use syntax features. These settings can reflect the contribution of the removing part.

4.3 Evaluation Metrics

Following previous work [11], we use macro Precision/Recall/F1 and Accuracy to measure the simile recognition results. Following previous work on simile interpretation and generation [4], we use Hit@1 to measure the multi-choice accuracy.

Table 5. Simile recognition results. The BERT-base (fine-tuned with MSP-modified SR train set) is the base model to do the significant test for our models (* means statistically significant with $p < 0.01$).

Model	Precision	Recall	F1	Accuracy
MSP-modified SR Test set				
BERT-base	0.7127	0.6981	0.6939	0.6996
Ours	**0.7904***	**0.7905***	**0.7905***	**0.7905***
(- SR training)	0.5000*	0.5000*	0.3768*	0.5000*
(- SI training)	0.7712*	0.7725*	0.7718*	0.7717*
(- SG training)	0.7774*	0.7801*	0.7781*	0.7779*
(- Semantic Dependency)	0.7822*	0.7805*	0.7836*	0.7821*
Our Proposed Test set				
BERT-base	0.4949	0.4963	0.4559	0.4922
Ours	**0.5419***	**0.5393***	**0.5332***	**0.5413***
(- SR training)	0.4927	0.4968	0.4179	0.5026
(- SI training)	0.5030*	0.5020*	0.4532*	0.4974*
(- SG training)	0.5152*	0.5136*	0.4985*	0.5110*
(- Semantic Dependency)	0.5325*	0.5284*	0.5114*	0.5256*

4.4 Implementation Details

Our model is implemented by PyTorch [16]. The implementations of the pre-trained models in this paper are all based on the public Pytorch implementation (https://github.com/huggingface/transformers). During the training, the maximum input length is set to 512. We use a single Tesla v100s GPU with 32 gb memory for experiments. The batch size is all set to 24. The model is optimized using the Adam optimizer with a learning rate of 5e−6. The learning rate is scheduled by a warm-up and linear decay. A dropout rate of 0.1 is applied for all linear transformation layers. The gradient clipping threshold is set as 10.0. Early stopping on the corresponding validation data is adopted as a regularization strategy. The training epochs are ∼3. For SI/SG testing on the new unseen set, if the masked position is a single word, we select the answer with the highest probability of the masked position; if there are multiple masked words, we encode the predicted words and the candidates into dense vectors with a sentence-transformer (https://www.huggingface.co/sentence_transformers/all-MiniLM-L6-v2). Then we compute the cosine similarity between the predicted words and each of the candidates. The candidate with the highest similarity is chosen as the answer.

5 Results and Analysis

In this section, we introduce the experimental results and provide our analysis of the results.

5.1 Simile Recognition

Table 5 shows the simile recognition results. The experiments are conducted on the MSP-modified SR test set and our new unseen test set.

Comparing with Baseline. The BERT-base model is fine-tuned with the MSP-modified SR train set and is tested with two test sets. One is the MSP-modified SR test set and the other is our new test set. We can see that on both test sets, our model performs better than the baselines. On the MSP-modified SR test set, our model surpasses BERT-base by around 7.8% on accuracy. On our proposed test set, our model outperforms BERT-base by around 4.9% on accuracy. On Macro Precision/Recall/F1, our model also outperforms the BERT-base model. The results show that our method not only can help PLM to use a more accurate simile knowledge but also perform better on a more difficult unseen test set. The results on the new test set are much lower than the MSP-modified SR test set, which indicates the new test set is much harder. Although our method helps the PLM to obtain a better performance on this new test set, there is still a lot of room to improve.

Ablation Study on SR. We also report the ablation study in Table 5. We can see that on both the MSP test set and the new test set, removing the key component of our model will cause declines. On the MSP test set, (- SR training) is exactly 50% because the model does not understand the SR task without the SR training. On the new test set, similar results are observed. The results are also around 50% and are not statistically significant.

On both test sets, (- SI training) performs worse than (- SG training). The results indicate that the SI fine-tuning task (recovering the masked property) is more useful than the SG fine-tuning task (recovering the masked vehicle) for the model to learn SR knowledge. It is because the shared property usually serves as the root of the semantic dependency tree. As shown in the (c) part of Fig. 1, the shared property connects most words in a simile sentence and the vehicle only connects a few words. When training with SI, the model learns more semantic relations between words than training with SG, so that the model can better leverage this semantic dependency knowledge for the SR task.

(- Semantic Dependency) causes more declines on the new test set (from 0.9–2.2% on all metrics) than on the MSP test set (from 0.7–1.0% on all metrics). It means the semantic dependency information helps the PLM to learn a more generic simile knowledge. This generic simile knowledge brings more gains in an unseen domain.

To sum up, experimental results on SR verify that 1) our method can explore more accurate and generic simile knowledge; 2) each fine-tuning task and the semantic dependency signal contributes to the performance.

Table 6. Simile interpretation and generation results (Hit@1) on MSD-En. The BERT-Probe is the base model to do the significant test for other models (* means statistically significant with $p < 0.01$).

Model	Interpretation	Generation
MSP-original SI Test set and MSP-modified SG Test set		
BERT-base (without fine-tuning)	0.7436	0.8155
BERT-Probe [8]	0.8015	0.8667
BERT-ANT [4]	0.8020	0.8675
Ours	**0.8101***	**0.8986***
(- SR training)	0.8006*	0.8819*
(- SI training)	0.7273*	0.8608*
(- SG training)	0.7832*	0.8113*
(- Semantic Dependency)	0.8089*	0.8799*
Our proposed Test set (the simile data)		
BERT-base (without fine-tuning)	0.5905	0.4510
BERT-Probe [8]	0.6454	0.5031
BERT-ANT [4]	0.5921	0.5094
Ours	**0.6142***	**0.5232***
(- SR training)	0.6084*	0.5189*
(- SI training)	0.5801*	0.4976*
(- SG training)	0.6025*	0.4888*
(- Semantic Dependency)	0.6031*	0.5022*

5.2 Simile Interpretation and Generation

Table 6 shows the simile interpretation and simile generation results. The SI task uses the MSP-original SI test set and our new test set. The SG task uses the MSP-modified SG test set and our new test set.

Comparing with Baselines. The first baseline is the BERT-base model without any fine-tuning. We can see that BERT-Probe performs better than BERT-base on both SI/SG tasks. The results are reasonable since BERT-Probe benefits from the fine-tuning of MSP-original/MSP-modified data on SI/SG tasks, respectively.

Different from the above two baselines, BERT-ANT is based on BERT-large and trained with a large corpus through Adjective-Noun mask Training. Benefiting from both a larger parameter size and the training process, BERT-ANT outperforms the BERT-Probe on both SI/SG tasks.

On the other hand, our model surpasses the strong BERT-ANT on both SI/SG even though our model uses BERT-base as the backbone. The results again verify that our method can enhance PLM with more accurate and generic simile knowledge.

The results on the new test set are still lower than the MSP test sets. One notable result is that the gap between results on the SG task is much larger than the gap on the SI task. The results show that the MSP-modified SG test set is easier than the MSP-original SI test set. The Hit@1 results are 89.86% and 81.01%, respectively. This may also be one of the reasons why SI training contributes more than SG training in Table 5. We can try constructing more difficult SG training data to improve the learning efficiency of our model.

Ablation Study on SI/SG. We also report the ablation study in Table 6. We can see that on both MSP test sets and the new test set, removing the training component of our model will cause declines.

On the MSP-original SI test set, (- SI training) causes ∼8.3% declines. On the new test set, (- SI training) only has ∼2.4% declines. The results are reasonable since the unseen test set is not as sensitive to the training data as the seen test set. A similar trend can be observed with the SG task. On the MSP-modified SG test set, (-SG training) causes ∼8.7% declines. On the new test set, (- SG training) only entails ∼3.4% declines.

On all test sets, (- SR training) only causes a little decline, which indicates that the SR fine-tuning contributes little to SI/SG tasks. This is different from the experimental results in Table 5, where SI/SG training contribute more to the SR task. How to leverage SR training to improve the SI/SG tasks requires further study.

Similar to the SR experiments, (- Semantic Dependency) causes more declines on the new test set (∼1.1% on SI and ∼2.1% on SG) than on MSP test sets (∼0.1% on SI and ∼1.9% on SG). The results mean the semantic dependency information helps more on an unseen set than the seen set, which is consistent with the results of the SR task.

To sum up, experimental results on SI/SG again verify that 1) our method can explore more accurate and generic simile knowledge; 2) each fine-tuning task and the semantic dependency signal have positive effects on the performance.

6 Conclusion

We propose a novel method to explore more accurate and generic simile knowledge from PLMs. We fine-tune PLM with three simile tasks (recognition, interpretation, and generation) to explore local simile knowledge between key simile components (tenor, shared property, vehicle). Then we use the semantic dependency feature for global simile knowledge among different examples. This global simile knowledge can help our model perform well across domains. Experiments with seen and unseen test sets verify the effectiveness of our method. Our exploring method may be useful for other NLP tasks that leverage knowledge from PLMs. Since our method does not need an expensive pre-training process, it may also be useful for leveraging more large-scaled PLMs. Future works include but are not limited to **1)** testing our method on other knowledge-intensive tasks; **2)** verifying whether our method can be transferred to auto-regressive-based PLMs.

Acknowledgements. This research project is supported by the National Natural Science Foundation of China (61872402), Science Foundation of Beijing Language and Culture University (supported by "the Fundamental Research Funds for the Central Universities") (18ZDJ03).

References

1. Birke, J., Sarkar, A.: A clustering approach for nearly unsupervised recognition of nonliteral language. In: McCarthy, D., Wintner, S. (eds.) EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, 3–7 April 2006, Trento, Italy. The Association for Computer Linguistics (2006). https://aclanthology.org/E06-1042/
2. Bizzoni, Y., Lappin, S.: Predicting human metaphor paraphrase judgments with deep neural networks. In: Klebanov, B.B., Shutova, E., Lichtenstein, P., Muresan, S., Leong, C.W. (eds.) Proceedings of the Workshop on Figurative Language Processing, Fig-Lang@NAACL-HLT 2018, New Orleans, Louisiana, 6 June 2018, pp. 45–55. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/W18-0906
3. Chakrabarty, T., Muresan, S., Peng, N.: Generating similes effortlessly like a pro: a style transfer approach for simile generation. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, 16–20 November 2020, pp. 6455–6469. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-main.524
4. Chen, W., et al.: Probing simile knowledge from pre-trained language models. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), ACL 2022, Dublin, Ireland, 22–27 May 2022, pp. 5875–5887. Association for Computational Linguistics (2022). https://doi.org/10.18653/v1/2022.acl-long.404
5. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, vol. 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019)
6. Dziri, N., Kamalloo, E., Mathewson, K.W., Zaïane, O.R.: Augmenting neural response generation with context-aware topical attention. CoRR abs/1811.01063 (2018). http://arxiv.org/abs/1811.01063
7. Dziri, N., Milton, S., Yu, M., Zaïane, O.R., Reddy, S.: On the origin of hallucinations in conversational models: is it the datasets or the models? In: Carpuat, M., de Marneffe, M., Ruíz, I.V.M. (eds.) Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, 10–15 July 2022, pp. 5271–5285. Association for Computational Linguistics (2022). https://doi.org/10.18653/v1/2022.naacl-main.387
8. He, Q., Cheng, S., Li, Z., Xie, R., Xiao, Y.: Can pre-trained language models interpret similes as smart as human? In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), ACL 2022, Dublin, Ireland, 22–27 May 2022, pp. 7875–7887. Association for Computational Linguistics (2022). https://doi.org/10.18653/v1/2022.acl-long.543

9. Lewis, M., et al.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020, pp. 7871–7880. Association for Computational Linguistics (2020). https://www.aclweb.org/anthology/2020.acl-main.703/
10. Li, Y., Lin, C., Guerin, F.: CM-GEN: a neural framework for Chinese metaphor generation with explicit context modelling. In: Calzolari, N., et al. (eds.) Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, 12–17 October 2022, pp. 6468–6479. International Committee on Computational Linguistics (2022). https://aclanthology.org/2022.coling-1.563
11. Liu, L., Hu, X., Song, W., Fu, R., Liu, T., Hu, G.: Neural multitask learning for simile recognition. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018, pp. 1543–1553. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1183
12. Liu, T., et al.: A token-level reference-free hallucination detection benchmark for free-form text generation. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), ACL 2022, Dublin, Ireland, 22–27 May 2022, pp. 6723–6737. Association for Computational Linguistics (2022). https://doi.org/10.18653/v1/2022.acl-long.464
13. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, 22–27 June 2014, Baltimore, MD, USA, System Demonstrations, pp. 55–60. The Association for Computer Linguistics (2014). https://doi.org/10.3115/v1/p14-5010
14. Mohler, M., Brunson, M., Rink, B., Tomlinson, M.T.: Introducing the LCC metaphor datasets. In: Calzolari, N., et al. (eds.) Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, 23–28 May 2016. European Language Resources Association (ELRA) (2016). http://www.lrec-conf.org/proceedings/lrec2016/summaries/1156.html
15. Niculae, V., Danescu-Niculescu-Mizil, C.: Brighter than gold: figurative language in user generated comparisons. In: Moschitti, A., Pang, B., Daelemans, W. (eds.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, 25–29 October 2014, Doha, Qatar, A Meeting of SIGDAT, a Special Interest Group of the ACL, pp. 2008–2018. ACL (2014). https://doi.org/10.3115/v1/d14-1215
16. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019 (December), vol. 32, pp. 8–14, 2019, Vancouver, BC, Canada, pp. 8024–8035 (2019). https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html
17. Paul, A.M.: Figurative language. In: Philosophy and Rhetoric, pp. 225–248 (1970)
18. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)

19. Shuster, K., Poff, S., Chen, M., Kiela, D., Weston, J.: Retrieval augmentation reduces hallucination in conversation. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 16–20 November 2021, pp. 3784–3803. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.findings-emnlp.320
20. Song, W., Guo, J., Fu, R., Liu, T., Liu, L.: A knowledge graph embedding approach for metaphor processing. IEEE ACM Trans. Audio Speech Lang. Process. **29**, 406–420 (2021). https://doi.org/10.1109/TASLP.2020.3040507
21. Speer, R., Chin, J., Havasi, C.: ConceptNet 5.5: an open multilingual graph of general knowledge. In: Singh, S., Markovitch, S. (eds.) Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4–9 February 2017, San Francisco, California, USA, pp. 4444–4451. AAAI Press (2017). http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972
22. Steen, G.: A Method for Linguistic Metaphor Identification: From MIP to MIPVU, vol. 14. John Benjamins Publishing, Amsterdam (2010)
23. Stowe, K., Beck, N., Gurevych, I.: Exploring metaphoric paraphrase generation. In: Bisazza, A., Abend, O. (eds.) Proceedings of the 25th Conference on Computational Natural Language Learning, CoNLL 2021, Online, 10–11 November 2021, pp. 323–336. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.conll-1.26
24. Su, C., Tian, J., Chen, Y.: Latent semantic similarity based interpretation of Chinese metaphors. Eng. Appl. Artif. Intell. **48**, 188–203 (2016). https://doi.org/10.1016/j.engappai.2015.10.014
25. Tsvetkov, Y., Boytsov, L., Gershman, A., Nyberg, E., Dyer, C.: Metaphor detection with cross-lingual model transfer. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, 22–27 June 2014, Baltimore, MD, USA, vol. 1: Long Papers, pp. 248–258. The Association for Computer Linguistics (2014). https://doi.org/10.3115/v1/p14-1024
26. Zhang, J., et al.: Writing polishment with simile: Task, dataset and A neural approach. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, 2–9 February 2021, pp. 14383–14392. AAAI Press (2021). https://ojs.aaai.org/index.php/AAAI/article/view/17691