It's Better to Teach Fishing than Giving a Fish: An Auto-Augmented Structure-aware Generative Model for Metaphor Detection

Huawen Feng, Qianli Ma
School of Computer Science and Engineering,
South China University of Technology, Guangzhou, China
541119578@qq.com, qianlima@scut.edu.cn

Abstract

Metaphor Detection aims to identify the metaphorical meaning of words in the sentence. Most existing work is discriminant models, which use the contextual semantic information extracted by transformers for classifications directly. Due to insufficient training data and corresponding paraphrases, recent methods focus on how to get external resources and utilize them to introduce more knowledge. Currently, contextual modeling and external data are two key issues in the field. In this paper, we propose **A**n **A**uto-**A**ugmented **S**tructure-aware generative model (**AAAS**) for metaphor detection, which transforms the classification task into a keywords-extraction task. Specifically, we propose the task of structure information extraction to allow the model to use the 'structural language' to describe the whole sentence. Furthermore, without any other external resources, we design a simple but effective auto-augmented method to expand the limited datasets. Experimental results show that **AAAS** obtains competitive results compared with state-of-the-art methods.

1 Introduction

Metaphors, representing abstract meanings of words rather than their basic meanings, are ubiquitous in our daily life (Lakoff and Johnson, 1980). For instance, in the sentence "The boxer's job is to ***bounce*** people who want to enter the club.", the verb ***bounce*** means "forcing somebody to leave", which is quite different from its basic meaning "moving up and down". As an abstract way of describing something by referring to something else, this language phenomenon draws extensive scholarly attention in linguistics. Hence, how to identify the metaphors has become a heated topic in NLP, with an aim to improve our understanding of natural language.

This challenging task requires sufficient data and ingenious designs based on linguistic knowledge. For years, linguisticians tended to use two

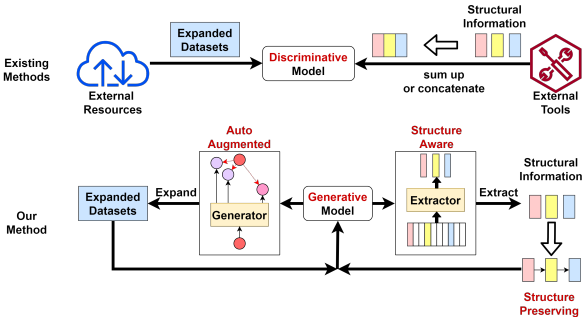

Figure 1: The comparison diagram of existing methods and our method.

metaphor identification procedures: Metaphor Identification Procedure (MIP) (Crisp et al., 2007) and Selectional Preference Violation (SPV) (Wilks, 2007). MIP identifies metaphorical words based on whether their contextual meanings are contrasted with their basic meanings. SPV identifies metaphorical words if the target word is distinctive in the context. Based on these two procedures, most existing methods tend to encode the whole sentence and extract the corresponding hidden state of the target word, which is then used as the contextual meaning for classifications (Gao et al., 2018). To get a better representation, various word embeddings (e.g., ELMO embedding (Liu et al., 2018)) and attention mechanisms are introduced into the input and structure of the model (Mao et al., 2019). However, due to the limited capacity of traditional encoders (e.g., Bi-LSTM, DNN), these methods cannot model the sophisticated meanings of target words in different contexts.

With the rapid development of transformers (Vaswani et al., 2017), many methods started to use various pre-trained language models to get better contextual representations. For example, DeepMet (Su et al., 2020) uses transformers to encode the global context and the local text context, respectively. At the same time, incorporating more linguistic knowledge while designing the model also

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 656–667
December 7-11, 2022

attracts scholarly attention. MelBERT (Choi et al., 2021), for instance, combines MIP and SPV with RoBERTa (Liu et al., 2019), a typical example of integrating linguistic knowledge with pre-trained models. Based on the concept that a metaphor is a conceptual mapping between the source domain and the target domain (Stowe et al., 2021), MrBERT (Song et al., 2021) extracts the subject and the object of the metaphor through the syntax parser to construct a contextual relation representation.

Moreover, owing to the similarity between metaphor detection, aspect-based sentiment analysis (Pontiki et al., 2016), and word sense disambiguation (Miller et al., 1994), some studies adopt multi-task learning to further improve the models' sensitivity to metaphorical words (Le et al., 2020; Stowe et al., 2021). Nevertheless, these methods rely on large amounts of data for fine-tuning. However, labeled data for metaphor detection is scarce due to labor-intensive and time-consuming labeling. Consequently, most of them have to employ transfer learning for better results.

Various external resources are mined to introduce extra knowledge to cope with the issue of insufficient data. CATE (Lin et al., 2021) downloads the corpora from Wikipedia and then generates pseudo-labels for training. MDGI (Wan et al., 2021) takes advantage of dictionary definitions to create the list of glosses, which facilitates the model's understanding of targets. In terms of the experimental results, these methods solve the issue of insufficient labeled data to a certain extent. However, external resources are still hard to access, and the whole training stage is time consuming.

To sum up, previous models are over-reliant on external resources and tools, but data is hard to obtain, and the training process takes too much time. Meanwhile, all of the existing methods are discriminant. Nearly all of them concatenate or add up the representation of contextual words directly and then input the result into the classifier, which may lose some essential connections between them. Even though some early methods are based on seq2seq models (Mao et al., 2019), they regard metaphor detection as a word-level classification (sequence labeling task) and ignore the linguistic structure.

The problems mentioned above motivated us to propose **A**n **A**uto-**A**ugmented **S**tructure-aware generative model (**AAAS**) for metaphor detection. Just as Figure 1 shows, almost all of the existing discriminant methods directly concatenate or sum up the contextual relation representation, which may lead to a loss of structural information. Given that, we adopted the generative approach, which models the structural information more accurately. Specifically, in the process of decoding, the decoder of the generative model took sequential relationships and interrelationships in contextual structure into consideration. In order to adapt the training process for application scenarios of the generative model, we designed a special keywords-extraction task for training. Considering that we can identify a metaphor by its subjects and objects in most circumstances, the task requires that the model summarizes the original sentence with structural terms. In other words, the model needs to describe the critical semantics of the whole sentence with subject, target, object, and classification results. As a result, the model itself can extract the structural information from the sentence, which makes it independent of external tools such as the syntax parser. In addition, we designed a simple but effective auto-augmented method based on the masked language model. The method can expand the dataset without any external resource, which fundamentally solves the problem of insufficient labeled data. To achieve a better performance, we added some structural rules to the expansion stage. In a word, we enhanced the model's capabilities so that it can extract structural information and expand datasets independently. Just as our title says, "**It's better to teach a man to fish than give him a fish.**"

In summary, the contributions of this paper are as follows: (1) Through a detailed analysis of the existing methods, we point out the problems in metaphor detection. (2) We propose an auto-augmented structure-aware generative model. To the best of our knowledge, it is the first time to apply the generative approach to metaphor detection and free the model from external resources. (3) We conduct experiments on several typical datasets for metaphor detection. Extensive analytical experiments show the effectiveness of both the generative model and the auto-augmented method in improving prediction performance, even compared with those relying on large-scale external resources.

2 Related Work

The work related to our method can be categorized into three types: Adopting multi-task learning, min-

ing the structural information, and introducing external resources.

Adopting multi-task learning Metaphor detection is quite similar to aspect-based sentiment analysis and word sense disambiguation, because they all require the model to classify data according to the target word and sentence. Through multi-task learning, the knowledge learned from auxiliary tasks (e.g., ABSA, WSD, and so on) can promote the training stage of the major task (MD) (Le et al., 2020; Stowe et al., 2021). Nevertheless, there are still some differences between these tasks. For example, metaphor words are usually identified according to the context, including their subjects and objects, whereas sentiment polarities are determined solely based on adjectives with strong emotions. Therefore, these similar tasks are not the most appropriate auxiliary tasks for metaphor detection.

Mining the structural information The study of metaphor generation (Stowe et al., 2021) indicates that a metaphor word is deemed to be a mapping between its source domain and target domain, which are closely related to the fixed group of the target word and its context (Lakoff and Johnson, 1980; Lakoff, 1993; Reddy, 1979). As a result, the linguistic structure is of great significance in identifying the metaphor word in a sentence. The contextual representations become more distinguishable based on subjects and objects extracted by the syntax parser (Chen and Manning, 2014). Then they are concatenated or added up to get a local or global representation for classifications (Song et al., 2021). This approach has two main drawbacks: (1) The syntax parser is prone to error, especially for long sentences. The wrong subjects and objects would hinder the prediction process. (2) The concatenated or summed contextual representations may lose sequential relationships and interrelationships.

Introducing external resources The current research focuses more on the external resources such as external corpora (Lin et al., 2021), external dictionaries (Wan et al., 2021), and so on. The other two types of methods also depend on the extra large-scale dataset for fine-tuning. However, most of the external resources are hard to collect, and the pre-processing process is extremely complicated. Worse still, the training stage takes too much time because of massive data.

3 Proposed Method

Metaphor detection requires determining whether the target in the sentence is a metaphor word. Given a sentence S consisting of n words $S = \{w_1, w_2, ..., w_n\}$ and the target w_i chosen from it, the task asks the model to predict a binary label $y \in \{Metaphor, Literal\}$. In this section, we propose **A**n **A**uto-**A**ugmented **S**tructure-aware generative model (**AAAS**) for metaphor detection. Firstly, we design an auto-augmented mechanism based on BERT (Devlin et al., 2018) to improve the model's performance when available data is limited. As for the main architecture of the model, we use a typical generative model, BART (Lewis et al., 2019), as our backbone, while the other generative models can also accommodate our architecture. Particularly, we design a decoder containing the pointer network (Vinyals et al., 2015) and the decoder of BART because the specially designed keywords-extraction task needs the words from the original sentence to construct the structural information.

3.1 Auto-augmented mechanism based on masked language model

The basic assumption of our design is based on the following principle (Actually, it is an imperfection of the BERT): For a mask word in the sentence, the most probable prediction given by BERT is usually a non-metaphor word (More detailed discussions are shown in Appendix A). The occurrence of this phenomenon could be attributable to the insufficient metaphor corpora in the pre-training stage of BERT. Our auto-augmented approach can cope with the defect.

Our method is illustrated in Figure 2. For a metaphorical sentence "The boxer's job is to ***bounce*** people who want to enter the club", if the target word "***bounce***" is masked, predicted, and replaced - "The boxer's job is to ***kill*** people who want to enter the club", the label will change into "Literal". However, if the other words are masked, predicted, and replaced, the labels will not be changed even though the semantics are slightly strange. If the original sentence is literal, the new sentences will always be literal no matter which word we mask because BERT can not predict a metaphor word. In this way, j words are randomly selected from each sentence and predicted, and then the top-k probable predictions are chosen to generate new sentences. j and k can be adjusted for specific datasets. In particular, to keep the structural infor-

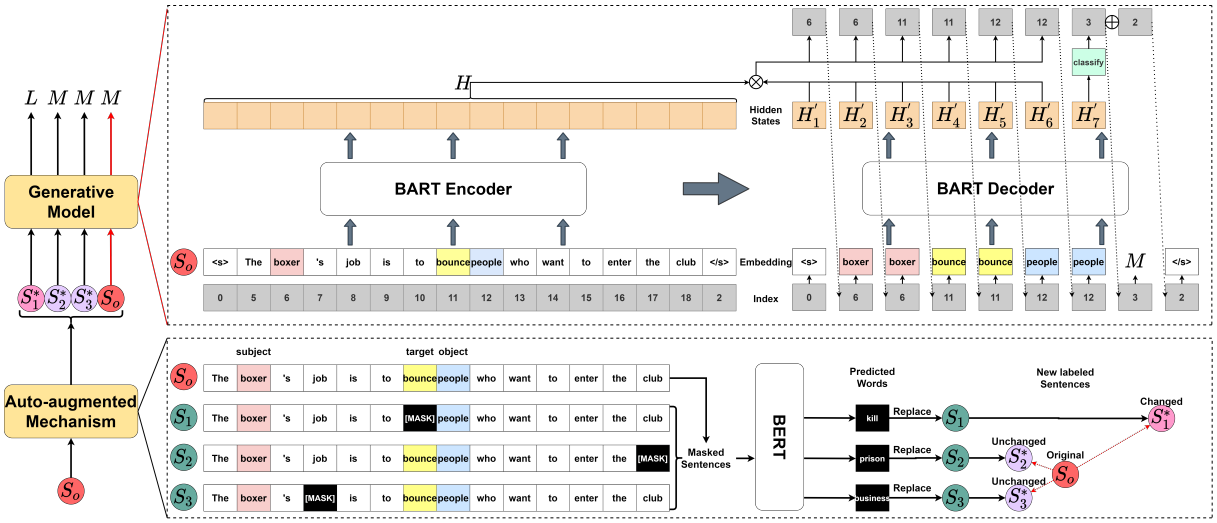

Figure 2: The diagram of our auto-augmented structure-aware generative model. After being expanded by the auto-augmented mechanism, the sentence is encoded by the encoder, which interacts with the decoder output to predict the next index of structural terms step by step. After extracting the subject, the target, and the object, the current decoder hidden state is multiplied with the embedding of "M" (Metaphor) and "L" (Literal) for a final classification.

mation, we avoid masking the subjects and objects of sentences.

In experiments, we observed that the smaller dataset needs larger values for j and k because they determine the size of expanded data. However, it is not to say that larger is better because the larger values can also result in more semantically incorrect sentences, so the expanded data may contain more noise. Whether to use the auto-augmented method and how to set (j, k) for it needs to be confirmed by Algorithm 1.

3.2 Structure-aware generative model

Our model consists of an encoder and a decoder with the pointer Network. As shown in Figure 2, given a sentence S, we first add the "<s>" and "</s>" to the beginning and end of the sentence because our encoder is based on BART. Special attention should be paid to the fact that the indexes corresponding to the words are not their positions. We set 0 for "<s>", 1 for "<pad>", 2 for "</s>", 3 for "M" (Metaphor), and 4 for "L" (Literal). Therefore, the indexes of words in the sentence are equal to their positions plus 5. To obtain the representation of the whole sentence, we get its embedding $X = \{x_1, x_2, ..., x_n\}$ and encode it by Equation 1:

$$\begin{aligned} X &= Embed_{Encoder}(S) \\ H &= Encoder(X) \end{aligned} \tag{1}$$

For the decoder part, the computing process is quite different from the encoder. As Equation 2 indicates, at the time step t, we get the current representation $H_t^{'} = \{h_1^{'}, h_2^{'}, ..., h_t^{'}\}$ determined by the current sentence $S_{i \leq t}^{'} = \{w_1^{'}, w_2^{'}, ..., w_t^{'}\}$ and encoder hidden states H, which is then multiplied with the encoder hidden states H according to the mechanism of the pointer network. The purpose of applying the pointer network is to get the probability distributions for the words from the original sentence because we need them to construct the structural representation.

$$\begin{aligned} X_{i \leq t}^{'} &= Embed_{Decoder}(S_{i \leq t}^{'}), t \in [1, 7] \\ H_t^{'} &= Decoder(X^{'}{}_{i \leq t}, H), t \in [1, 7] \\ P &= Softmax(HH_t^{'}), t \in [1, 6] \end{aligned} \tag{2}$$

3.3 Special keywords-extraction task

We design a particular keywords-extraction task to adapt the training process for the generative model. The task requires the model to summarize the whole sentence with the structural terms. Specifically, the model needs to find the subject, the target, and the object in sequence and determine whether the target is a metaphor word.

During the training stage, we design a seq2seq loss function. After getting the probability distributions $\hat{P} = \{\hat{p}_1, \hat{p}_2, ..., \hat{p}_7\}$ for a batch of sentences, the loss function is calculated as follows:

$$Loss_{total} = Loss_{classify} + \gamma Loss_{extract} \tag{3}$$

where γ is a hyperparameter that controls the strength of extracting structural information. Since extracting structural information is the auxiliary task but not the main task, γ is chosen from the range $[0, 1]$. The $Loss_{classify}$ is the cross-entropy of ground-truth labels $\hat{Y}$ and the probability results p_7:

$$Loss_{classify} = CrossEntropy(\hat{Y}, p_7) \quad (4)$$

And the $Loss_{extract}$ consists of the losses of extracting subjects, targets, and objects:

$$\begin{aligned} Loss_{subject} &= CrossEntropy(start_{subject}, p_1) \\ &+CrossEntropy(end_{subject}, p_2) \\ Loss_{target} &= CrossEntropy(start_{target}, p_3) \\ &+CrossEntropy(end_{target}, p_4) \\ Loss_{object} &= CrossEntropy(start_{object}, p_5) \\ &+CrossEntropy(end_{object}, p_6) \end{aligned} \quad (5)$$

The $CrossEntropy$ mentioned above is:

$$CrossEntropy(P, Q) = \frac{1}{M}\sum_{m=1}^{M} P_m \log Q_m \quad (6)$$

where M is the number of samples, P_m and Q_m are the ground-truth labels and predicted output for the m-th sample respectively.

The structural information in our train sets is extracted by the syntax parser (Chen and Manning, 2014), similar to MrBERT (Song et al., 2021). More parsing rules are introduced for more accurate results, but some noise still remains in the parsing output, which will interfere with the final classification. That is one of the reasons why we do not use parsing results directly while inferring. Moreover, there are some sentences without subjects or objects. A special token "<null>" is appended to the sentence, and the model is asked to predict its index when that exceptional case occurs.

The above part of Figure 2 indicates an example of the inference stage. The length of the expected output sequence (the structural representation) is fixed as 7, including the start index and end index of the subject, target, and object, and a final classification result. At the beginning of the inference phase, we input the beginning of sequence token "<s>" and the decoder output "6" - the start index of the subject ($start_{subject}$). The word "boxer" is then generated according to the index and appended to the inputs. Next, we input "<s> boxer" to predict the end index of the subject ($end_{subject}$). Similarly, we get "<s> boxer boxer" and predict the start index of the target. In this way, all the indexes of the structural words are extracted, and the current hidden state turns out to be $H_7^{'}$. However, the decoder hidden state $H_7^{'}$ is not multiplied with H this time. Instead, we multiply it with the vectors of "M" (Metaphor) and "L" (Literal) in $Embed_{Decoder}$. Therefore, the decoder can only predict the label from $\{Metaphor, Literal\}$ because of the restriction. Finally, we add "</s>" to the end of the sequence to finish the inference stage.

3.4 Searching for best settings

In experiments, we find that the decoder input's content and order impact the results. For the example in Figure 2, we can input only a token - "<s>", but we can also input "<s> boxer boxer", which means we can offer more ancillary information while inferring. Taken to an extreme, we can input "<s> boxer boxer bounce bounce people people" and make the model classify the data directly. On the other hand, order is also vital for prediction. The classification result of "subject-target-object-label" can be quite different from "target-subject-object-label". Figure 2 only indicates one scenario and the detailed results will be discussed in Section 4.4. Apart from that, our auto-augmented mechanism needs j and k to expand datasets, and their values are also supposed to be appropriate. Therefore, how to choose the best pattern (content and order), j, and k is crucial for the results.

As Algorithm 1 shows, we confirm the best settings through D_{val}. It's worth noting that $template$ is determined earlier than j_{best} and k_{best}, due to the finite number of possible permutations of the structural words.

4 Experiments

Compared with existing models, we tried AAAS on several metaphor detection tasks. Experimental results demonstrate that AAAS consistently achieves strong performance on all datasets, which outperforms all of the state-of-the-art baselines methods in terms of accuracy and F1-score. In this section, we attempt to answer the following questions: **RQ1:** Does AAAS perform better than existing methods? **RQ2:** Is AAAS still excellent while removing its auto-augmented mechanism? **RQ3:** How do the pattern, j, k, and γ affect the results?

Algorithm 1: Searching algorithm

Input: train set D_{train}, validation set D_{val}, pre-trained generative model $f(\cdot;\theta;pattern)$, auto-augmented method $g(\cdot;j,k)$, maximum tolerance J, and maximum sampling number K.

foreach pattern **do**
  Train the model on D_{train} and update θ using Adam.
  Get F1-score on D_{val}.
end
Get the best pattern $template$ for highest F1-score.
for $j = 0, 1, ..., J$ **do**
  for $k = 1, ..., K$ **do**
    Expand D_{train} to get $D_{train}^{expand} = g(D_{train}; j, k)$.
    Train the model $f(\cdot;\theta;template)$ on D_{train}^{expand} and update θ using Adam.
    Get F1-score on D_{val}.
  end
end
Get the best values j_{best} and k_{best} of j and k for highest F1-score.
return $template, j_{best}, k_{best}$

4.1 Experimental settings

4.1.1 Datasets and preprocessing

To evaluate the effectiveness of our model, we conduct experiments on three widely-used datasets: (1) **MOH-X** (Mohammad et al., 2016) is a small dataset, and only a single target verb is annotated in each sentence. (2) **TroFi** (Birke and Sarkar, 2006) is also a verb metaphor detection dataset, including sentences from the 1987-89 Wall Street Journal Corpus Release 1. (3) **VUA** (Steen et al., 2010) is a large dataset divided into a train set, a validation set, and a test set. It is used by the NAACL-2018 Metaphor Shared Task and consists of two main tracks: VERB and All_POS metaphor detection.

The details of the three datasets are listed in Table 1. According to the common search algorithms of generative method, we adopt the beam search with a beam width of 4. We select the best pattern, j, and k for each dataset by Algorithm 1 and their influences are discussed in Section 4.4. More detailed settings are shown in Appendix B. The code will be made publicly available.

Dataset	Targets	Metaphors	Sentences	Avglen
MOH-X	647	48.7%	647	8.0
Trofi	3737	43.5%	3737	28.3
VUA_VERB$_{train}$	15,516	27.9%	7,479	20.2
VUA_VERB$_{val}$	1,724	26.9%	1,541	25.0
VUA_VERB$_{test}$	5,873	30.0%	2,694	18.6
VUA_All_POS$_{train}$	116,622	11.2%	6,323	18.4
VUA_All_POS$_{train}$	38,628	11.6%	1,550	24.9
VUA_All_POS$_{train}$	50,175	12.4%	2,694	18.6

Table 1: Detailed dataset statistics.

4.1.2 Baselines

We compare our models with current strong baselines, including:

RNN_CLS, **RNN_SEQ_ELMo** and **RNN_SEQ_BERT** (Gao et al., 2018): Use various embedding (e.g., Glove embedding (Pennington et al., 2014), ELMo embedding, and BERT embedding) and their combinations to get better contextual representation. **RNN_HG** and **RNN_MHCA** (Mao et al., 2019): Both of them adopt sequence labeling. According to MIP and SPV, they concatenate the embedding and hidden states and utilize multi-head attention to capture better contextual information. **MUL_GCN** (Le et al., 2020): Introduce word sense disambiguation as an auxiliary task and use GCN (Heidari et al., 2022) to capture structural contexts. **BERT+MWE_GCN** (Rohanian et al., 2020): Get targets' syntactic dependencies by an attention-based GCN to further capture multiword expressions. **DeepMet** (Su et al., 2020): Encode global and local context by RoBERTa (Liu et al., 2019) and combine them. **MelBERT** (Choi et al., 2021): Use RoBERTa as the backbone to get contextual and literal meaning while incorporating both MIP and SPV. **MrBERT** (Song et al., 2021): Extract the structures of sentences by the syntax parser and concatenate their contextual representations. **CATE** (Lin et al., 2021): Trained on external dataset downloaded from Wikipedia, CATE uses contrastive learning to enhance the model's self-training to get better pseudo-labels.

4.2 RQ1: Performances compared with existing methods

Table 2 shows the experimental results of AAAS compared with others on three benchmarks. The overall results indicate the effectiveness of our AAAS. We can find that the performance of AAAS is excellent on all datasets, which exceeds existing models in terms of accuracy and F1-score, especially on the small dataset - MOH-X.

Models	MOH-X(10-fold)				Trofi(10-fold)				VUA_VERB				VUA_All_POS			
	Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1	Acc	P	R	F1
RNN_CLS	78.5	75.3	84.3	79.1	73.7	68.7	74.6	72.0	69.1	53.4	65.6	58.9	-	-	-	-
RNN_SEQ_ELMo	77.2	79.1	73.5	75.6	74.6	70.7	71.6	71.1	81.4	68.2	71.3	69.7	93.1	71.6	73.6	72.6
RNN_SEQ_BERT	78.1	75.1	81.8	78.2	73.4	70.3	67.1	68.7	80.7	66.7	71.5	69.0	92.9	71.5	71.9	71.7
RNN_HG	79.7	79.7	79.8	79.8	74.9	67.4	**77.8**	72.2	82.1	69.3	72.3	70.8	93.6	71.8	76.3	74.0
RNN_MHCA	79.8	77.5	83.1	80.0	75.2	68.6	76.8	72.4	81.8	66.3	**75.2**	70.5	93.8	73.0	75.7	74.3
MUL_GCN	79.9	79.7	80.5	79.6	76.4	73.1	73.6	73.2	83.2	72.5	70.9	71.7	93.8	74.8	75.5	75.1
BERT+MWE_GCN	80.5	80.0	80.4	80.2	73.5	73.8	71.8	72.8	-	-	-	-	-	-	-	-
DeepMet	-	-	-	-	-	-	-	-	-	79.5	70.8	74.9	-	82.0	71.3	76.3
MelBERT	-	-	-	-	-	-	-	-	-	78.7	72.9	75.7	-	80.1	76.9	78.5
MrBERT	84.9	84.1	85.6	84.2	76.7	73.9	72.1	72.9	**86.4**	80.8	71.5	75.9	94.7	**82.7**	72.5	77.2
CATE	85.2	85.7	84.6	84.7	**77.7**	**74.4**	74.8	74.5	85.8	78.1	73.2	75.6	94.8	79.3	**78.8**	79.0
AAAS	**87.5**	**89.5**	**85.2**	**87.0**	**77.7**	72.5	77.5	**74.8**	**86.4**	**81.6**	71.1	**76.0**	**95.2**	81.6	77.4	**79.4**

Table 2: Experimental results on three metaphor detection benchmarks. The best result is in **bold**.

It is noteworthy that almost all the transformer-based models (e.g., MUL_GCN, MelBERT, MrBERT, and CATE) proposed recently needed external datasets for fine-tuning, transferring learning, or multi-task learning. However, AAAS does not rely on any other extra resources and still achieves excellent results. Compared with large-scale external corpora, our auto-augmented mechanism can generate smaller-scale but high-quality data, saving a lot of training time and fundamentally solving the problem of insufficient data.

4.3 RQ2: Is AAAS still excellent while removing its auto-augmented mechanism?

Models	MOH-X(10-fold)			
	Acc	P	R	F1
RNN_CLS	78.5	75.3	84.3	79.1
RNN_SEQ_ELMo	77.2	79.1	73.5	75.6
RNN_SEQ_BERT	78.1	75.1	81.8	78.2
RNN_HG	79.7	79.7	79.8	79.8
RNN_MHCA	79.8	77.5	83.1	80.0
MrBERT	84.9^{2*}	84.1^{3*}	85.6^{2*}	84.2^{3*}
CATE	85.2^{1*}	85.7^{1*}	84.6^{3*}	84.7^{2*}
AAAS w/o AA	84.2^{3*}	84.6^{2*}	87.1^{1*}	85.6^{1*}

Models	Trofi(10-fold)			
	Acc	P	R	F1
RNN_CLS	73.7	68.7	74.6	72.0
RNN_SEQ_ELMo	74.6	70.7	71.6	71.1
RNN_SEQ_BERT	73.4	70.3	67.1	68.7
RNN_HG	74.9	67.4	77.8^{1*}	72.2
RNN_MHCA	75.2	68.6	76.8^{2*}	72.4
MrBERT	76.7^{2*}	73.9^{2*}	72.1	72.9^{3*}
CATE	77.7^{1*}	74.4^{1*}	74.8	74.5^{1*}
AAAS w/o AA	76.1^{3*}	71.2^{3*}	76.4^{3*}	73.5^{2*}

Table 3: Experimental results after removing the auto-augmented mechanism. The best results is in red, the second is in orange, and the third is in blue.

In order to verify the effectiveness of the generative method, we remove the auto-augmented approach and train our model only on the original datasets. The results are reported in Table 3. Even without the auto-augmented method and expanded datasets, our structural-aware generative model can obtain superior or competitive results compared with previous models dependent on large-scale datasets, proving our generative method's validity. Specifically, in the case of insufficient data, the structure-aware generative architecture gets a better contextual representation than existing discriminant models.

However, compared with complete AAAS, the results do decrease significantly. The phenomenon demonstrates our auto-augmented mechanism helps improve the performance of the model, especially on the small dataset - MOH-X. Expansion based on the auto-augmented approach generates better data much less costly than those relying on external resources and tools. The expanded dataset is similar to the original one in that the auto-augmented method generates new sentences based on the original ones, but the external datasets introduced by previous methods and original ones do not belong to the same schema at all, which may be a reason why our auto-augmented mechanism performs better than other extension methods.

4.4 RQ3: How do the pattern, j, k, and γ affect the results?

As explained in Section 3.4, the pattern we choose substantially impacts the final results. Therefore, we design Algorithm 1 to obtain the best one. As shown in Table 4, for MOH-X, the content of the decoder input does little to influence the final results. The results of $S^* - T - O - L$ and $S^* - T^* - O - L$ are even the same, which indi-

cates that our structure-aware generative model can extract structural information through training, so good results can be obtained with or without auxiliary input. However, for Trofi, $T^* - S - O - L$ shows the greatest accuracy while the patterns with the subject or the object have poorer performance. We guess this is because the syntax parser performs badly in long sentences, so the structural information generated contains some noise, which affects the final classification. By the way, we have also tried other patterns like $S^* - S^* - T^* - T^* - O^* - O^* - L$, but the pattern is more time-costing (requires twice as much time as the short patterns need) and not obviously better than other patterns. Hence, we don't choose long patterns like it.

Pattern	MOH-X(10-fold)			
	Acc	P	R	F1
$S - T - O - L$	84.88	87.80	82.14	84.37
$S^* - T - O - L$	**85.23**	88.60	81.54	84.54
$S^* - T^* - O - L$	**85.23**	88.60	81.54	84.54
$S^* - T^* - O^* - L$	**85.23**	88.44	81.84	84.58
$T - S - O - L$	85.05	85.49	84.91	85.07
$T^* - S - O - L$	85.05	84.36	87.08	85.41
$T^* - S^* - O - L$	85.06	84.59	86.75	85.37
$T^* - S^* - O^* - L$	84.22	84.63	87.08	85.55
Pattern	Trofi(10-fold)			
	Acc	P	R	F1
$S - T - O - L$	75.83	71.68	73.80	72.21
$S^* - T - O - L$	75.02	70.47	75.21	72.56
$S^* - T^* - O - L$	74.72	69.61	76.09	72.57
$S^* - T^* - O^* - L$	76.10	71.18	76.41	73.50
$T - S - O - L$	75.32	69.15	78.72	73.50
$T^* - S - O - L$	**76.18**	72.79	72.31	72.48
$T^* - S^* - O - L$	75.34	69.16	78.79	73.53
$T^* - S^* - O^* - L$	74.39	68.23	79.22	73.15
Pattern	MOH-X(10-fold)			
	Acc	P	R	F1
$S^* - S^* - T^* - T^* - O^* - O^* - L$	85.56	86.24	84.97	85.44

Table 4: Experimental results of different patterns. We remove the auto-augmented mechanism and keep two decimals here to observe the results more clearly. Here we show two orders and four patterns with varying numbers of input elements for each order (e.g., $S^* - T^* - O - L$ means the order of inferring is "subject-target-object-label" and we input the subject and target before predicting). The best accuracy is in **bold**.

For j and k, the values we choose determine the size of the expanded dataset. As explained in Section 3.1, it is not to suggest that larger is better because larger values can result in more semantically incorrect sentences. The influence of j and k is shown in Figure 3. Since we use Algorithm 1 to select better settings, the F1 score of AAAS without the auto-augmented mechanism is the lower bound (the black dashed line). For different values of k, the performances reach the peaks at different values of j, and then decrease due to the injection of too much noise. Overall, the larger k fits the smaller j better. We speculate that it is because there is a limit to how much noise the model can accommodate. The performance will be improved if the auto-augmented method expands the data within this limit. Beyond this limit, it will play a limited role.

As for γ, its value controls the strength of extracting structural information. From Figure 3 we can find that, with the increase of γ, the F1 score first rises to the peak when γ is 0.01 and then declines with fluctuations. The phenomenon indicates that the keywords-extraction task can help improve the model's performance. Still, it will be less effective if we focus too much on it and ignore the main task (Metaphor Identification). More detailed discussions are shown in Appendix C.

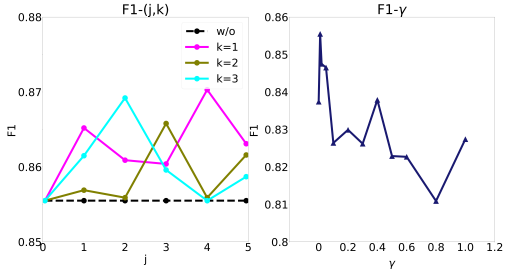

Figure 3: The chart of the fluctuations of F1 when we change the values of j, k, and γ on MOH-X(10-fold). We use $T^* - S^* - O^* - L$ and $T^* - S - O - L$ as the pattern for the two experiments here, respectively. And we remove the auto-augmented mechanism for the one of F1-γ.

5 Conclusion

This paper summarizes existing metaphor detection methods, indicating that almost all of them are discriminant models and rely on external corpora and tools. We propose a structure-aware generative model with an auto-augmented mechanism to solve the problems. We conduct massive experiments on several metaphor detection datasets and achieve remarkable performance. The experimental results demonstrate the effectiveness of the gener-

ative method for capturing sequential relationships and interrelationships and the auto-augmented approach for solving the problem of insufficient data. We expect our work will direct more scholarly attention to generative models for metaphor detection and data auto-augmentation methods elaborately designed for insufficient labeled data.

Limitations

In this work, we first propose a cost-free solution to the problem of insufficient labeled data. We then propose a generative model to capture better sequential relationships and interrelationships. The auto-augmented method solves the problem of labor-intensive and time-consuming labeling. And the structure-aware model avoids the loss of structural information. However, searching for the pattern, j, and k takes too much time. Additionally, the training of the generative model requires a lot of GPU resources. Overall, AAAS needs a lot of computing resources to obtain better results.

Acknowledgements

The work described in this paper was partially funded by the National Natural Science Foundation of China (Grant Nos. 62272173, 61872148), the Natural Science Foundation of Guangdong Province (Grant Nos. 2022A1515010179, 2019A1515010768). We thank Zhenxi Lin, Haibin Chen for providing suggestion on the paper.

References

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language.

D. Chen and C. Manning. 2014. A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1763–1773, Online. Association for Computational Linguistics.

Peter Crisp, Raymond Gibbs, Alice Deignan, Graham Low, Gerard Steen, Lynne Cameron, Elena Semino, Joe Grady, Alan Cienki, Zoltán Kövecses, and The Group. 2007. Mip: A method for identifying metaphorically used words in discourse. Metaphor and Symbol, 22.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. pages 607–613.

Negar Heidari, Lukas Hedegaard, and Alexandros Iosifidis. 2022. Graph convolutional networks, pages 71–99.

G. Lakoff and M. Johnson. 1980. Metaphors we live by. Ethics, 19(2):426–435.

George Lakoff. 1993. The contemporary theory of metaphor.

Duong Minh Le, My Thai, and Thien Huu Nguyen. 2020. Multi-task learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 8139–8146, Online.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. CoRR, abs/1910.13461.

Zhenxi Lin, Qianli Ma, Jiangyue Yan, and Jieyu Chen. 2021. CATE: A contrastive pre-trained model for metaphor detection with semi-supervised learning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 3888–3898, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liyuan Liu, Xiang Ren, Jingbo Shang, Xiaotao Gu, Jian Peng, and Jiawei Han. 2018. Efficient contextualized representation: Language model pruning for sequence labeling. In EMNLP, pages 1215–1225.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. CoRR, abs/1907.11692.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3888–3898, Florence, Italy. Association for Computational Linguistics.

G. A. Miller, M. Chodorow, S. Landes, C. Leacock, and R. G. Thomas. 1994. Using a semantic concordance for sense identification. In Human Language

Technology, Proceedings of a Workshop held at Plainsboro, New Jerey, USA, March 8-11, 1994.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. pages 23–33.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. volume 14, pages 1532–1543.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeny Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 Task 5: Aspect Based Sentiment Analysis. In International Workshop on Semantic Evaluation, pages 19 – 30, San Diego, United States.

M. J. Reddy. 1979. The conduit metaphor: A case of frame conflict in our language about language. Metaphor and Thought.

Omid Rohanian, Marek Rei, Shiva Taslimipoor, and Le Ha. 2020. Verbal multiword expressions for identification of metaphor.

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021. Verb metaphor detection via contextual relation learning. pages 4240–4251.

Gerard Steen, Lettie Dorst, J. Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. A method for linguistic metaphor identification: From MIP to MIPVU.

Kevin Stowe, Tuhin Chakrabarty, Nanyun Peng, Smaranda Muresan, and Iryna Gurevych. 2021. Metaphor generation with conceptual mappings. CoRR, abs/2106.01228.

Chuandong Su, Fumiyo Fukumoto, Xiaoxi Huang, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. Deepmet: A reading comprehension paradigm for token-level metaphor detection. pages 30–39.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc.

Hai Wan, Jinxia Lin, J.F. Du, Dawei Shen, and Manrong Zhang. 2021. Enhancing metaphor detection by gloss-based interpretations. pages 1971–1981.

Yorick Wilks. 2007. A Preferential, Pattern-Seeking, Semantics for Natural Language Inference, pages 83–102. Springer Netherlands, Dordrecht.

A Discussions about the Design of the Auto-augmented Mechanism

As explained in Section 3.1, for a mask word in the sentence, the most probable prediction given by BERT is usually a non-metaphor word. In this way, for a metaphorical sentence (Labeled as "M"), if the target word (metaphor) is masked, predicted, and replaced, the label will change into "L" (Literal). On the contrary, if the other words (e.g., articles, prepositions, adjectives, possessive pronouns, and so on) are masked, predicted, and replaced, the labels will change. As for a literal sentence (Labeled as "L"), the label is not supposed to change no matter which word we mask because BERT can not predict a metaphor word. Some examples are shown in Table 5. The above four examples can prove our assumption.

It is noteworthy that we avoid masking the subjects and objects of sentences to keep the structural information because the process of replacing them is not controllable. For example, there is a metaphorical sentence - "She ***drowned*** in the trouble.". If the object ("trouble") is masked, a new object ("water") will be predicted, which makes the metaphorical sentence literal. Similar errors can also happen in literal sentences. For the literal sentence - "I can not ***digest*** the milk.", if "milk" is masked, "information" will be predicted, and the label is supposed to change into "M". However, masking subjects and objects does not necessarily result in a change in labels (e.g., labels will not change if we turn "She" into "I", "He", "We", and any other pronouns). Whether the labels should be changed is hard to decide because we do not anticipate all scenarios. As a result, we skip this for subjects and objects to ensure accuracy of expansion.

Nevertheless, sentences generated by the auto-augmented approach are not entirely appropriate. There can be a few strange semantics and wrong combinations of phrases, but these minor mistakes do not affect the understanding of sentence meanings and the identification of metaphors at all.

B Detailed Settings

The detailed settings of experiments are shown in Table 6. We set γ as 0.01 and conducted experi-

Original Sentences & Labels	Masked Sentences	Generated Sentences & Labels
Fire had devoured our home. & M	Fire had **[MASK]** our home. Fire had devoured **[MASK]** home.	Fire had **destroyed** our home. & L Fire had devoured **her** home. & M
He absorbed the knowledge or beliefs of his tribe. & M	He **[MASK]** the knowledge or beliefs of his tribe. He absorbed the knowledge **[MASK]** beliefs of his tribe. He absorbed the knowledge or **[MASK]** of his tribe.	He **has** the knowledge or beliefs of his tribe. & L He absorbed the knowledge **and** beliefs of his tribe. & M He absorbed the knowledge or **wisdom** of his tribe. & M
The rain water drains into this big vat. & L	The rain water **[MASK]** into this big vat. The **[MASK]** water drains into this big vat. The rain water drains into this **[MASK]** vat.	The rain water **went** into this big vat. & L The **hot** water drains into this big vat. & L The rain water drains into this **large** vat. & L
The truck dumped the garbage in the street. & L	The truck **[MASK]** the garbage in the street. **[MASK]** truck dumped the garbage in the street. The truck dumped the garbage in the **[MASK]**.	The truck **and** the garbage in the street. & L **A** truck dumped the garbage in the street. & L The truck dumped the garbage in the **ditch**. & L
She drowned in the trouble. & M	She **[MASK]** in the trouble. **[MASK]** drowned in the trouble. She drowned in the **[MASK]**.	She **was** in the trouble. & L **I** drowned in the trouble. & M She drowned in the **water**. & L
I can not digest the milk. & L	I can not **[MASK]** the milk. I can **[MASK]** digest the milk. I can not digest the **[MASK]**.	I can not **drink** the milk. & L I can **hardly** digest the milk. & L I can not digest the **information**. & M

Table 5: Experimental results of the auto-augmented mechanism.

Dataset	γ	LR_{max}	LR_{max}	Warmup_Steps	Total_Steps	Batch_Size	Device
MOH-X(10-fold)	0.01	1e-4	1e-6	50	500	64	GeForce RTX 3090
Trofi(10-fold)	0.01	1e-4	1e-7	50	1000	50	GeForce RTX 3090
VUA_VERB	0.01	2e-6	1e-7	600	1000	32	GeForce RTX 3090
VUA_All_POS	0.01	2e-6	1e-6	800	1000	50	GeForce RTX 3090

Table 6: Detailed experimental settings.

ments on GeForce RTX 3090 for all datasets. Almost all the sentences in Trofi consist of several clauses, which are much longer than MOH-X. By analyzing datasets, we find the meanings of targets are usually decided by the clause where they are located. Considering that, we split the long sentences in Trofi by commas and only reserve the clauses containing targets, which helps remove unwanted information and reduce the demands for large GPU resources. As for MOH-X, we skip this because most of its sentences are short enough.

C More Discussions about the Keywords-extraction Task

Models	MOH-X(10-fold)			
	Acc	P	R	F1
$T^* - L$	83.58	85.06	81.47	82.71

Table 7: Experimental results of $T^* - L$ (vanilla BART's performance). That is the most basic pattern for Metaphor Identification.

As we explained in Section 1, some existing methods adopt multi-task learning, but there are still some differences between these auxiliary tasks and Metaphor Identification. For instance, for Aspect-Based Sentiment Analysis, the classification results are solely based on adjectives with strong emotions. Therefore, the key to dealing

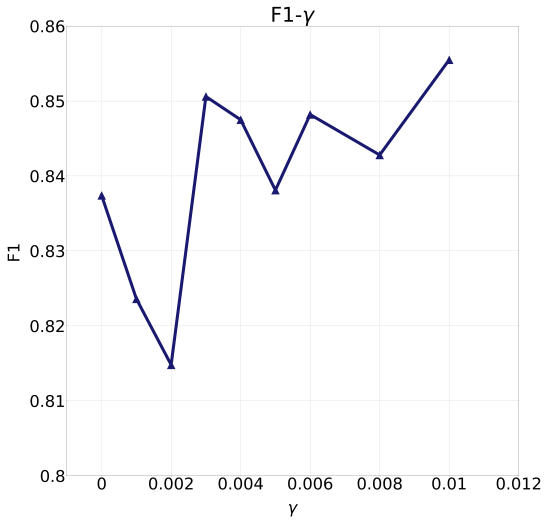

Figure 4: The chart of the fluctuations of F1 when we change the value γ in the range $[0, 0.01]$ on MOH-X(10-fold).

with the task is to find the adjectives corresponding to the given aspect term. However, things are different for Metaphor Identification. It is impossible to determine whether the target is a metaphor word by only one word. Given that, we propose the keywords-extraction task, which allows the model to extract the subjects, targets, and objects, because we can identify a metaphor by its subjects and objects in most circumstances. Extensive experiments

show that our keywords-extraction task is more effective than other existing auxiliary tasks.

In Section 4.4, we talked about the fluctuations of F1 when we change the values of γ in the range $[0, 1]$, finding that the keywords-extraction task can help improve the model's performance, and it works best when γ is 0.01. To verify this value further, we narrow the range to $[0, 0.01]$, as shown in Figure 4, and come to the same conclusion as the previous one.

To prove the effectiveness of the auxiliary task more fully, we design a pattern - $T^* - L$, which is the most basic input for Metaphor Identification. According to Table 4 and Table 7, we can conclude that the extraction process of structural terms improves the classification accuracy significantly.

D Experiments Based on the Smaller Backbone

Most of the previous methods use BERT-base and BERT-large as baselines. We have tried to replace BERT-base with BERT-large while reproducing them but could not always get better results. We guess larger BERT may not behave better. Considering that, we compare our model with their best results they published.

In experiments, we use BART-large as our backbone. Actually, the size of BART and BERT are similar at the same level. For example, BART-large consists of a 12-layer encoder and a 12-layer decoder while BERT-large has 24 layers. We also conduct experiments based on BART-base and the results are still competitive compared with the previous works. To conclude, larger BERT is not suitable for all methods, but larger BART is suitable for our generative method. Our structure-aware model based on BART-base is still effective enough.

Datasets	$S^* - T^* - O^* - L$			
	Acc	P	R	F1
MOH-X(10-fold)	85.5	85.4	84.0	84.5
w/o AA	84.2	85.8	81.8	83.6
Trofi(10-fold)	76.5	71.4	76.6	73.9
w/o AA	74.9	69.7	77.4	73.0

Table 8: Experimental results based on BART-base.