Knowledge-guided Metaphor Detection using Generative Language Models.

Anonymous ACL submission

Abstract

Metaphor detection is an important and challenging task in the field of natural language process. Previous works typically directly encodes the target word and the entire text using pretrained models to extract corresponding semantic information. This approach may not only be affected by the multiple meanings of the target word but also may be influenced by irrelevant information in the context, leading the model to extract inaccurate or biased information and thus make incorrect judgments. To address these issues, we propose **K**nowledge-**G**uided **M**etaphor **D**etection using generative language models(KGMD). By using generative language models to understand and perform synonymous replacements for the basic and semantic information of the target word, the model is guided to better utilize semantic information and capture contextual relationships. Additionally, we designed a framework that combines large generative models with pretrained bidirectional language models. This framework not only takes full advantage of the rich world knowledge and excellent semantic understanding capabilities of large generative models but also retains the efficient training performance and cost advantages of bidirectional language models. Experimental results verify that our method achieves significant performance gains compared to the SOTA methods in metaphor detection.

1 Introduction

In our daily lives, metaphor is a widely used concept(Lakoff and Johnson, 2008). People use concrete and easily understandable things to describe abstract concepts or things that are difficult to comprehend. For example, "break the ice" uses "ice" to metaphorically represent a newly formed unfamiliar and stiff interpersonal relationship. The existence of metaphors allows people to intuitively understand one thing according to certain similar characteristics of another, occupying an important position in linguistics and cognitive science.

The task of metaphor detection(Gao et al., 2018) is widely recognized in the field of nature language process, but it is also a very challenging task. Metaphor detection involves determining whether a word in a given text contains a metaphor(Do Dinh and Gurevych, 2016). Current metaphor detection work is primarily based on two principles: Metaphor Identification Procedure (MIP)(Conneau et al., 2017; Reimers, 2019) and Selectional Preference Violation (SPV)(Haagsma and Bjerva, 2016; Do Dinh et al., 2018). MIP identifies metaphors by comparing the basic meaning of a word with its contextual meaning; SPV identifies metaphors by looking for abnormal relationships between a word and its less common collocations.

In the field of metaphor detection, using large language models (LLMs)(Kasneci et al., 2023) is not a new idea. Many efforts were dedicated to utilizing the advantages of large models in semantic understanding(Xu et al., 2024), capturing the source and target domains of metaphors(Wachowiak and Gromann, 2023), providing conceptual explanations of metaphors(Yang et al., 2024), or designing metaphor datasets suitable for large models to evaluate their applications in this field(Tong et al., 2024). These efforts are mostly in the evaluation and initial stages, and there remains much to be explored before true application can be realized. Additionally, the cost of using large language models cannot be ignored(Long et al., 2024). On the other hand, traditional metaphor detection methods involve directly encoding the target word and the entire text or introducing external knowledge to extract corresponding semantic information(Gao et al., 2018). This approach not only may be affected by the multiple meanings of a word but also might lose important contextual information during the encoding computation process(Turney et al., 2011). These

shortcomings are major bottlenecks limiting the performance of models in metaphor identification. Inspired by the recent excellent performance of large generative language models in semantic understanding and bidirectional language models in semantic computation, we use a framework combining generative and bidirectional models to address the bottlenecks of metaphor detection in extracting the basic and contextual meanings.

In this paper, we proposed **K**nowledge-**G**uided **M**etaphor **D**etection using generative language models(KGMD). Through well-designed prompts and chain-of-thought, KGMD guides large generative models to provide synonymous explanations and contextual replacements for target words. This approach enables direct and accurate semantic and contextual understanding for the overall framework. At the same time, KGMD uses generative models to provide multiple interpretations and synonymous replacements for the same target word based on contextual context, effectively avoiding potential biases, and without altering the original semantics. Through the creative combination of large generative models and small bidirectional models, KGMD can leverage the advantages of generative models in semantic understanding while also benefiting from the efficient performance of small bidirectional models. By designing a module that uses generative models separately, our framework exhibits good flexibility and scalability. It is compatible with metaphor identification models that use different principles and allows for the selection of the type and size of the generative model based on actual needs.

Distinguished from methods that introduce structured and manually annotated knowledge, and those that rely solely on generative models, our approach fundamentally leverages the internal knowledge of generative models through a bidirectional model. This means we do not need to introduce complex externally annotated knowledge. At the same time, our method leverages the efficiency and ease of training of smaller bidirectional models, enabling the entire framework to effectively self-learn the threshold for metaphor discrimination. This approach avoids the inherent drawback of generative models being less suitable for classification tasks.

Our main contributions are as follows:

- We have identified the excellent performance of large generative models in contextual understanding and designed efficient prompts and chain-of-thought to guide these models in making accurate interpretations and applications of metaphors. This method generates synonymous semantic information suitable for encoding, computation, and training with bidirectional models.

- For the first time, we proposed a framework for metaphor detection that creatively combines generative models and bidirectional models, leveraging the accurate contextual understanding capabilities of generative models and the efficient performance of bidirectional models, while also being highly scalable.

- Experiments show that our model outperforms all previous models, demonstrating significant performance improvements over original results. Additionally, we provide detailed ablation studies and analyses to illustrate the effectiveness of our model.

2 Related Work

2.1 Metaphor Detection

Early work on metaphor identification was primarily feature-based.(Shutova and Sun, 2013; Shutova et al., 2010) and often utilized methods based on Recurrent Neural Networks(Gao et al., 2018; Mao et al., 2019). These methods largely depend on manually annotated knowledge. Moreover, due to the limitations of RNNs themselves, these methods also have significant limitations in extracting contextual information. Current work is primarily based on two principles: Metaphor Identification Procedure (MIP)(Group, 2007; Steen et al., 2010) and Selectional Preference Violation (SPV)(Wilks, 1975; Wilks, 1978). SPV posits that if the target word does not match its preferred collocations in the current context, there is a high likelihood that the word is a metaphor. For example, although "kill time" is a set phrase, the basic semantics of "kill" prefer pairing with biological entities; thus, this usage violates the preferences of the word "kill". MIP posits that if the basic meaning of the target word differs significantly from its meaning in the current context, then the word is considered to be a metaphor. By combining these two principles with transformer-based models, significant progress has been made in the field of metaphor identification. Zhang and Liu (2022) models MIP and SPV as two semantic matching tasks, combining methods

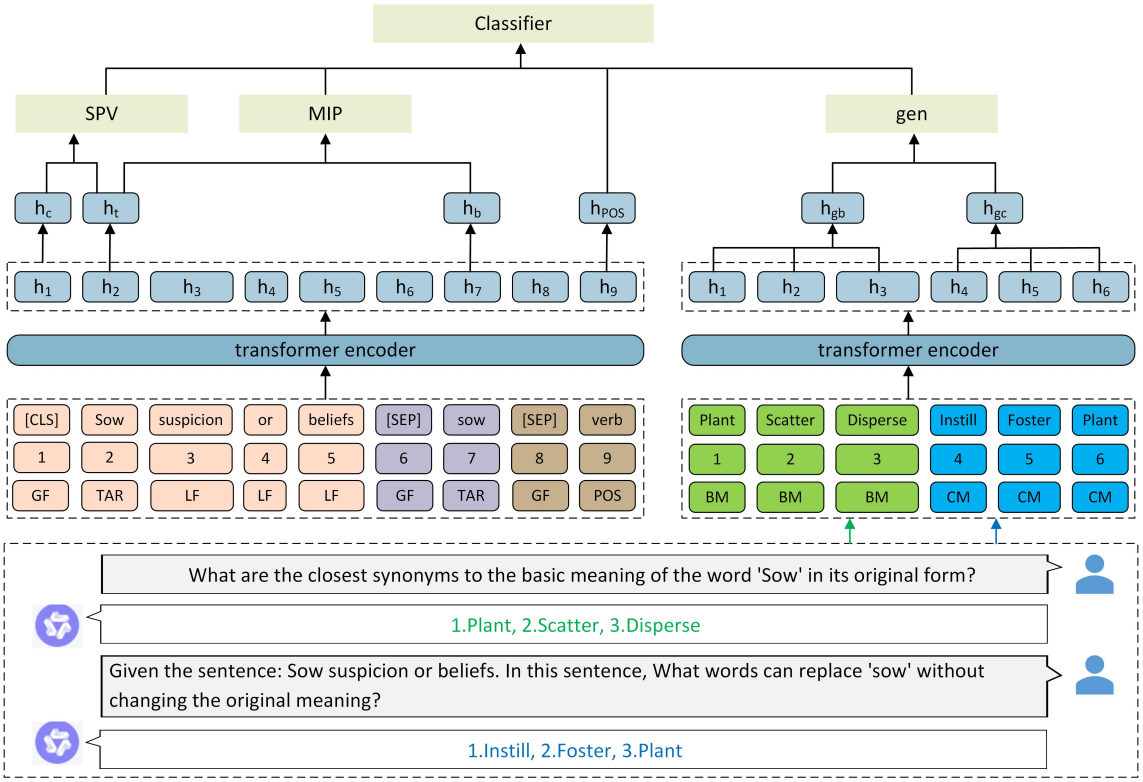

Figure 1: The overview of our framework. The three layers of input represent: token embedding, position embedding and feature embedding. GF,TAR,LF,POS,BM,CM denote global feature, target word, local feature, POS feature, Basic Meaning and Context Meaning.

based on these two principles with pre-trained models for the first time. Wan et al. (2021) and Su et al. (2021) uses manually annotated annotations to interpret the target word. Tian et al. (2023) model the similarity between the source domain and the target domain, addressing the gap in the field of metaphor detection regarding similarity. Li et al. (2023) and Wang et al. (2023) improve the model's limitations in contextual understanding by introducing external knowledge contained in other models as auxiliary components. Our method is also based on the two principles of MIP and SPV and uses the pre-trained model BERT(Kenton and Toutanova, 2019) as the backbone of the model.

2.2 Metaphor Detection with LLMs

Noting the superior advantages of large language models in semantic understanding, many efforts have also attempted to introduce LLMs into metaphor detection work. Xu et al. (2024) approaches leveraged the advantages of LLMs in multimodal understanding to interpret images containing metaphors and then align these interpretations with image features, achieving significant progress in the field of multimodal metaphor detection. Wachowiak and Gromann (2023) used reasonably designed prompts for few-shot learning, leveraging generative models to identify the source and target domains of metaphors. Yang et al. (2024) used ChatGPT to construct a generalized list of target word combinations and detected metaphors through the entailment relationships between the original sentence and the items in the list. Tong et al. (2024) addressed the issue of current metaphor datasets containing a large number of conventional metaphors and lacking explanations by proposing a metaphor detection dataset suitable for large models. Our method is a deeper exploration based on these works.

3 Method

We proposed a novel metaphor detection framework, KGMD. Figure 1 illustrates the main architecture of KGMD, which consists of two parts: the basic semantic encoder (Section 3.1) and the generative model semantic encoder (Section 3.2). The basic semantic encoder uses a pre-trained bidirectional model to explicitly learn the metaphor representations in the context(Radford et al., 2019). The generative model semantic encoder utilizes

generative models to generate sequences of substitutable synonyms that reflect the dual meanings of the metaphor.

3.1 Basic Semantic Encoder

The basic semantic encoder's role is to encode the input sequence containing the target word, extracting its basic meaning representation, contextual meaning representation, contextual representation, and part-of-speech (POS) representation. The SPV and MIP are then used to perform initial computations on the first three representations, resulting in h_{SPV} and h_{MIP}.

Formally, given an input sequence $S = (w_0, w_1, ...w_n)$ containing the target word w_i, we first restructure the input sequence into the following form:

$$S_1 = (w_0, w_1...w_n.w_i.POS) \quad (1)$$

After processing by the encoder, the beginning of the input sequence S_1 will be prefixed with a special start character [CLS], and special separator characters [SEP] will be inserted between w_n and w_i, as well as between w_i and POS. POS is the part of speech of the target word w_i (such as adj for adjective, adv for adverb, etc.). This input sequence, after being encoded by BERT, yields context-dependent vectors:

$$\mathrm{H} = \mathrm{BERT}(\mathrm{S}_1) = (\mathrm{h}_0, \mathrm{h}_1...\mathrm{h}_\mathrm{m}) \quad (2)$$

Generally, we denote $h_p \sim h_q$ as the hidden layer vectors corresponding to the target word w_i at the front of sequence S_1, and $h_v \sim h_s$ as the hidden layer vectors corresponding to the target word w_i at the back of sequence S_1. We use their respective averages as the extracted contextual meaning vector h_t and the basic meaning vector h_b:

$$h_t = \frac{1}{q-p}\sum_{i=p}^{q} h_i, h_b = \frac{1}{s-v}\sum_{i=v}^{s} h_i \quad (3)$$

Similarly, we use the average value of the input sequence as the contextual meaning vector h_s, and the hidden layer vector corresponding to the POS is denoted as h_{POS}.

MIP detects metaphors by comparing the contextual meaning and the surrounding context meaning of the target word. SPV detects metaphors by comparing the contextual meaning and the basic meaning of the target word. Following the settings of MISNET(Zhang and Liu, 2022), we calculate h_t, h_b and h_s as new vectors h_{SPV} and h_{MIP}:

$$h_{MIP}{=}W_{MIP}^{\top}[h_t; h_b; |h_t{-}h_b|;h_t{*}h_b]{+}b_{MIP} \quad (4)$$

$$h_{SPV}{=}W_{SPV}^{\top}[h_c; h_t; |h_c{-}h_t|;h_c{*}h_t]{+}b_{SPV} \quad (5)$$

where $[\cdot]$ is a readout method. $|\cdot|$ denotes the absolute value. ; indicates concatenation, and $*$ denotes the Hadamard product. W and b represent the weights and biases of the corresponding layer respectively. This method retains the information of the original concatenation method while also maximizing the utilization of differential information.

3.2 Generative Model Semantic Encoder

While using the original combination of MIP and SPV principles, our framework additionally incorporates a component that aids semantic understanding with the assistance of a generative large model. The core method of this component involves designing reasonable prompts and chain-of-thought to deconstruct the basic semantics and contextual semantics of the target word and perform synonymous replacements. Finally, through a pre-trained BERT model, the synonyms and replaceable phrases generated by the generative model are used as basic semantics and contextual meanings, and then participate in the calculations.

This method can be divided into three steps:

- First, given the target word w_i alone, the generative model understands its basic meaning and generates multiple synonyms that are most similar to its basic meaning.
- Then, given the input sequence S containing the target word and historical information, the generative model generates multiple substitutable synonyms or phrases that are consistent with the current context of the target word.
- Finally, after filtering out unreasonable outputs, the outputs of the first two steps are fed into the pre-trained BERT model as basic and contextual meanings, respectively.

Following the above steps, we construct a two-step prompt chain-of-thought and a third step for information processing.

STEP1. To avoid contextual influence, we only provide the target word w_i, guiding the generative

model to generate multiple synonyms that are closest to the basic meaning of the target word. This process can be modeled as follows:

$$\mathrm{b_1, b_2 ... b_k = ex(GM(w_i, Question1))} \quad (6)$$

where GM represents the generative model used, ex represents the method of extracting key information from the generated content, and $\mathrm{b_i}$ represents the extracted synonym or synonym phrase. To avoid the influence of polysemy and words with multiple parts of speech, we guide the generative model to focus on the most basic meaning of the target word in its original form. The design of Question1 is as follows:

Question1: What are the closest synonyms to the basic meaning of the word '{target_word}' in its original form?

Figure 2: Prompt of Question1

STEP2. We provide the input S containing the target word and prompt the generative model to generate possible synonymous replacements for the target word without altering the contextual meaning. Note that the generation here builds upon the historical information from the initial interaction, guiding the model to respond along the lines of contextual synonyms. This process can be modeled as follows:

$$\mathrm{c_1, c_2 ... c_k = ex(GM(S, b_1 ... b_k, Question2))} \quad (7)$$

where $\mathrm{c_i}$ is the substitution of the target word generated without changing the meaning of the context. Combining the content generated in the first step, our aim is to ensure that the model's generated content refers to the basic meaning in terms of semantics, corresponding to the similarity between the metaphor and the literal term. At the same time, the generated content should connect to the entire contextual environment, reflecting the inconsistency between the metaphor and the literal term. Therefore, the specific design of Question 2 is as follows:

Question2: Given the sentence:{sentence} In this sentence, What words can replace '{target_word}' without changing the original meaning?

Figure 3: Prompt of Question2

STEP3. We filter out the overly long outputs based on the length of the generated content, and we also use manual inspection to remove any outputs that are clearly semantically incorrect. The obtained sequences of basic meanings and contextual meanings are fed into the pre-trained BERT model:

$$h_{b1}, h_{b2} ... h_{bm} = \mathrm{BERT(b_1, b_2 ... b_k)} \quad (8)$$

$$h_{c1}, h_{c2} ... h_{cn} = \mathrm{BERT(c_1, c_2 ... c_k)} \quad (9)$$

To avoid the biases inherent in the generative model itself and errors that may occur during the generation process, we use the average of multiple outputs as the generated basic meaning and contextual meaning:

$$h_{gb} = \frac{1}{m}\sum_{i=1}^{m} h_{bi}, h_{gc} = \frac{1}{n}\sum_{i=1}^{n} h_{bi} \quad (10)$$

Where h_{gb} and h_{gc} represent the hidden vectors of the basic meaning and contextual meaning obtained from the generative model, respectively. These two hidden vectors are merged using the same algorithm as described in section 3.1, resulting in h_{gen}. Finally, the obtained h_{gen} is combined with h_{MIP}, h_{SPV} and h_{POS} derived from section 3.1 through a linear layer to compute the output:

$$\hat{y} = \sigma(L(h_{MIP}; h_{SPV}; h_{POS}; h_{gen})) \quad (11)$$

Where σ denotes the softmax function, L represents the linear layer operation, and ; indicates concatenation. Finally, the binary cross-entropy function is used to calculate the loss between the output $\hat{y}$ and the label y, which trains the entire framework except for the generative model:

$$L = -\sum_{i=0}^{N} y_i \log \hat{y}_i - (1 - y_i) \log (1 - \hat{y}_i) \quad (12)$$

4 Experiments

4.1 Dataset

Our experiments were primarily conducted on the following four datasets: **(1)** VUA ALL(Steen et al., 2010), the largest dataset where target words include all parts of speech. **(2)** VUA Verb(Steen et al., 2010), a dataset that includes only the portion of VUA ALL where the target words are verbs. **(3)** MOH-X(Mohammad et al., 2016), a dataset that includes only verbs in terms of part of speech but

Model	VUA All				VUA Verb				MOH-X			
	Pre	Recall	F1	Acc	Pre	Recall	F1	Acc	Pre	Recall	F1	Acc
RNN_ELMo	71.6	73.6	72.6	93.1	68.2	71.3	69.7	81.4	79.1	73.5	75.6	77.2
RNN_BERT	71.5	71.9	71.7	92.9	66.7	71.5	69.0	80.7	75.1	81.8	78.2	78.1
RNN_HG	71.8	76.3	74.0	93.6	69.3	72.3	70.8	82.1	79.7	79.8	79.8	79.7
RNN_MHCA	73.0	75.7	74.3	93.8	66.3	75.2	70.5	81.8	77.5	83.1	80.0	79.8
MUL_GCN	74.8	75.5	75.1	93.8	72.5	70.9	71.7	83.2	79.7	80.5	79.6	79.9
RoBERTa_SEQ	80.4	74.9	77.5	-	79.2	69.8	74.2	-	-	-	-	-
DeepMet	82.0	71.3	76.3	-	79.5	70.8	74.9	-	-	-	-	-
MelBERT	80.1	76.9	78.5	-	78.7	72.9	75.7	-	-	-	-	-
MrBERT	**82.7**	72.5	77.2	94.7	**80.8**	71.5	75.9	**86.4**	80.0	**85.1**	82.1	81.9
MisNet	78.9	**77.8**	78.3	95.0	74.0	73.5	73.7	85.9	**81.5**	82.4	81.8	82.2
KGMD	80.8	76.9	**78.8**	**95.2**	73.7	**78.9**	**76.3**	86.3	80.6	84.5	**82.3**	**82.5**

Table 1: Results on VUA All, VUA Verb, and MOH-X. Best in **bold** and second best in italic underlined.

places more emphasis on polysemy and ambiguous metaphors. **(4)** TroFi(Birke and Sarkar, 2006, 2007), a real-world dataset, taken from the corpus of the Wall Street Journal from 1987-1989. The detailed information for the four datasets is shown in Table 2.

Dataset	#Sent.	#Target	%Met.	Avg.Len
VUA All(tr)	6,323	116,622	11.19	18.4
VUA All(val)	1,550	38,628	11.62	24.9
VUA All(te)	2,694	50,175	12.44	18.6
VUA Verb(tr)	7,479	15,516	27.90	20.2
VUA Verb(val)	1,541	1,724	26.91	25.0
VUA Verb(te)	2,694	5,873	29.98	18.6
MOH-X	647	647	48.69	8.0
TroFi	3,737	3,737	43.54	28.3

Table 2: Datasets information. #Sent: Number of sentences. #Target: Number of target words. %Met.: Percentage of metaphors. Avg.Len: Average sentence length

4.2 Baselines

(1) **RNN-ELMo**(Gao et al., 2018) and **RNN-BERT**(Mao et al., 2019): Models that combine with other pre-trained models on the basis of sequence labeling to integrate word semantic encoding. (2) **RNN-HG** and **RNN-MHCA**(Mao et al., 2019): The former uses the MIP principle, combines with other pre-trained models, and offers different encoding for word semantics; the latter uses the SPV principle to capture the differences in basic meaning and contextual meaning of words from different attention perspectives. (3) **MULGCN**(Le et al., 2020): A multi-task learning approach for metaphor detection that considers both word sense disambiguation and dependency relations together. (4) **RobertaSEQ**(Leong et al., 2020): A sequence labeling model using RoBERTa that calculates the probability of metaphor for each token in a sentence. (5) **DeepMet**(Su et al., 2020): A model that effectively utilizes query features and reformulates the metaphor detection task as a reading comprehension task. (6) **MelBert**(Choi et al., 2021): A method that simultaneously uses both the MIP and SPV principles. (7) **MrBert**(Song et al., 2021): A method based on relationships between words, which achieves metaphor detection by encoding these relationships and using pre-trained models for computation. (8) **MisNet**(Zhang and Liu, 2022): A method that improves upon MelBert, achieving more precise extraction of the basic meaning of target words.

4.3 Implementation Details

Our base pre-trained model uses the BERT model provided by Hugging Face, and the generative model uses the llama7b model provided by Hugging Face. During fine-tuning, the parameters of the generative model remain unchanged, and only the parameters of the BERT model are fine-tuned. We dynamically select the number of training epochs and the learning rate based on the size of the dataset and the used generative large model, and we also implemented a warmup for the learning rate. All baseline experiments were conducted on a single NVIDIA RTX 4090 GPU. For detailed information, please refer to Appendix A.

5 Results and Analysis

5.1 Overall Results

Table 1 shows that our method generally outperforms existing baselines. On the VUA All and MOH-X datasets, our method achieves higher F1 scores and accuracy compared to all previous methods, including the original state-of-the-art methods

MisNet and MrBert. Specifically, our method improves the F1 score by 0.5 and 0.2 on these datasets. Compared to methods using RNN, our method improves the F1 score by 6.2 and 6.7. On the VUA Verb dataset, our method also achieves performance that is either the same or similar to SOTA methods. In particular, the F1 score improved by 0.4, and the accuracy was only 0.1 lower. The experimental results demonstrate that KGMD can fully leverage the potential of generative models in semantic understanding, showing better performance in situations rich in various contexts, even when the number of samples is relatively insufficient. Although the smallest generative LLaMA model with only 7B parameters was used, it still outperforms previous SOTA methods. Although KGMD did not achieve the best performance in Precision (Pre) and Recall, its overall performance was higher than other methods. We speculate that this is because KGMD integrates the advantages of different modules, resulting in a trade-off where emphasizing performance in one aspect leads to a slight loss. Furthermore, only the parameters of the basic semantic encoder need to be fine-tuned, without the need for additional fine-tuning of the generative model.

5.2 Zero-shot Transfer on TroFi

We saved the model from the training epoch that performed best on VUA All dataset and directly used it to test on TroFi, in order to assess the generalization capabilities of KGMD. The results are shown in Table 3. In terms of F1 score, KGMD is 0.2 higher than MrBERT and 0.9 higher than MelBERT. In terms of accuracy, KGMD is 1.1 higher than MrBERT. The experiments show that KGMD also significantly leads other methods in terms of generalization capability. The generative model provides semantic understanding information that is lacking in ordinary methods. By encoding this information, our method achieves notable generalization ability.

Model	**TroFi(Zero-shot)**			
	Pre.	Recall	F1	Acc.
RoBERTa_SEQ	53.6	70.1	60.7	-
DeepMet	53.7	72.9	61.7	-
MelBERT	53.4	<u>74.1</u>	62.0	-
MrBERT	<u>53.8</u>	**75.0**	<u>62.7</u>	<u>61.1</u>
KGMD	**54.9**	73.5	**62.9**	**62.2**

Table 3: Zero-shot transfer results on TroFi dataset.

5.3 Ablation Study

Our ablation studies consist of three parts: experiments to investigate the impact of the number k of generated synonyms and contextual replacements, experiments to explore the influence of each component within our framework, and experiments to examine the effect of different sizes and structures of generative large models on the experimental results.

In the first experiment, we set $k = 1, 2, 3, 5$ respectively and conducted experiments under the same settings using the MOH-X dataset. The experimental results are shown in the Table 4. When the number of generated synonyms is small, the experimental results tend to be better because generative models usually output the most reasonable result first. However, this also makes the results more susceptible to bias. As the number of generated contents increases, the experimental results are best at $k = 3$. After that, the results start to decline. At $k = 5$, the F1 score drops by 2.0, and the accuracy drops by 2.2. This indicates that more generated content is not necessarily better, as an excess of contextually unrelated synonyms can reduce the influence of correct synonyms due to semantic loss during averaging.

Ablation	Pre.	Recall	F1	Acc.
$k = 1$	82.0	82.0	81.7	82.3
$k = 2$	81.0	**83.5**	81.9	82.5
$k = 3$	**84.5**	82.7	**83.2**	**84.1**
$k = 5$	81.7	81.4	81.2	81.9

Table 4: Effectiveness study of the number of generated content k on MOH-X dataset.

In the second experiment, we removed each of the MIP, SPV, and generative model components separately and conducted experiments under the same settings using the VUA Verb dataset. The experimental results are shown in Table 5. In the absence of any one of these components, the performance of the overall framework significantly decreases. When the SPV component is missing, the F1 score drops by 7.2 because it becomes impossible to compare the extracted basic semantic information with the contextual semantic information. When the MIP component is missing, the F1 score drops by 1.4; and when the generative model is missing, the F1 score also drops by 1.6, indicating that both the MIP and generative model play an indispensable role in the entire framework. Only in

its complete form does our method achieve its best performance.

Ablation	Pre.	Recall	F1	Acc.
-MIP	**74.0**	73.7	73.9	86.0
-SPV	62.9	74.3	68.1	81.3
-GEN	**74.0**	73.5	73.7	85.9
KGMD	72.0	**78.9**	**75.3**	**86.1**

Table 5: Effectiveness study of framework components on VUA Verb dataset.

In the third experiment, we investigated the impact of the size and structure of the generative models on the experimental results. Since we used a small generative model in base experiments, to demonstrate that our method can be generalized to generative large models of different magnitudes, we conducted experiments using LLaMA7B, LLaMA13B, Qwen-Turbo, GLM-4, and GPT3.5-Turbo on the MOH-X dataset. The experimental results are shown in Table 6. Overall, as the size of the model parameters increases, there is a noticeable improvement in the experimental results. Among them, the experiment using GPT3.5 shows the most significant improvement, while other models with similar parameter counts (such as Qwen-Turbo and GLM-4) also yield good results. The experimental results indicate that increasing the parameter count of larger models does improve the performance of KGMD. However, considering factors such as cost and task difficulty, using a smaller generative model is a more cost-effective choice.

5.4 Case Study

To more clearly illustrate the role of KGMD in metaphor detection, we selected several exemplary cases that form a case study. We use bold to mark the target word. In the following example:"She **radiates** happiness." Here, the basic meaning of the target word radiates is:"Emit (energy, especially light or heat) in the form of rays or waves." But in this context, the word radiates is used to express the display and infectiousness of a person's emotions. Our framework generates the following basic meanings for it: "Shines, Glows, Beams," and the following contextual meanings: "Exudes, Expresses, Emanates." It can be seen that KGMD effectively distinguishes between the basic meaning and the contextual meaning, accurately capturing the abstract connotations and successfully detecting this metaphor.

Model	Pre.	Recall	F1	Acc.
llama7B	80.6	**84.5**	82.3	82.5
llama13B	81.6	82.5	81.9	82.3
Qwen-Turbo	80.9	83.9	82.1	82.4
GLM-4	81.7	83.9	82.2	82.3
GPT3.5-Turbo	**84.5**	82.7	**83.2**	**84.1**

Table 6: Effectiveness study of model types and sizes on MOH-X dataset.

When the generative model component is missing, the extraction of semantics by the pre-trained model alone is inaccurate. For example, consider the sentence: "The war **inflated** the economy." Here, the contextual meaning of the target word "inflated" is quite special, referring to inflation as an economic issue rather than physical expansion. Without the generative model component, the framework failed to identify this metaphor, but it succeeded in the complete setup.

However, in cases involving more complex and less commonly used metaphorical words, our method still has room for improvement. For example, consider the phrase "**paste** the sign on the wall." In this sentence, "paste" is not a metaphor because it is used in its basic meaning: "to fix or attach something using adhesive." However, KGMD captured the basic meaning of "paste" as "Adhesive, Glue, Cement," and the contextual meaning as "Stick, Affix, Attach." Regarding the explanation of more complex constructions involving multiple parts of speech, our framework still has limitations.

6 Conclusion

In this paper, we propose a novel framework for metaphor detection called KGMD. By combining generative models with bidirectional pre-trained models, KGMD can fully exploit the advantages of generative models in semantic understanding while also retaining the efficiency benefits of bidirectional pre-trained models during training. With the help of the knowledge provided by generative models, we address the challenge of accurately extracting contextual information in the field of metaphor detection, achieving competitive results on four widely-used metaphor detection datasets. In summary, our work not only explores the potential of generative models in the field of metaphor detection but also achieves an effective integration of large and small models, paving the way for future research.

7 Limitations

Our work has some limitations. First, due to the costs associated with using generative models, we have limited exploration of larger models and have not validated the impact of increased model parameter sizes across all datasets. Additionally, we have only utilized four different architectures of generative models. Additionally, our experiments are based on the assumption that the theories of metaphor applies to all types of metaphors. However, in actual language use, the types of metaphors employed are very diverse, making it difficult to generalize them all under one umbrella. In the process of translating metaphor theories into model structures, simplifications and trade-offs are inevitably made. We leave these to a future study.

References

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European chapter of the association for computational linguistics*, pages 329–336.

Julia Birke and Anoop Sarkar. 2007. Active learning for the identification of nonliteral language. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 21–28, Rochester, New York. Association for Computational Linguistics.

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.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Erik-Lân Do Dinh and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 28–33.

Erik-Lân Do Dinh, Hannah Wieland, and Iryna Gurevych. 2018. Weeding out conventionalized metaphors: A corpus of novel metaphor annotations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1424.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. *arXiv preprint arXiv:1808.09653*.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol*, 22(1):1–39.

Hessel Haagsma and Johannes Bjerva. 2016. Detecting novel metaphor using selectional preference information. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 10–17.

Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota.

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

Duong Le, My Thai, and Thien Nguyen. 2020. Multi-task learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8139–8146. AAAI Press.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 VUA and TOEFL metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

Yucheng Li, Shun Wang, Chenghua Lin, Frank Guerin, and Loïc Barrault. 2023. Framebert: Conceptual metaphor detection with frame embedding learning. *arXiv preprint arXiv:2302.04834*.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 11065–11082. Association for Computational Linguistics.

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.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the fifth joint conference on lexical and computational semantics*, pages 23–33.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ekaterina Shutova and Lin Sun. 2013. Unsupervised metaphor identification using hierarchical graph factorization clustering. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 978–988.

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

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021. Verb metaphor detection via contextual relation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4240–4251, Online. Association for Computational Linguistics.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Chang Su, Kechun Wu, and Yijiang Chen. 2021. Enhanced metaphor detection via incorporation of external knowledge based on linguistic theories. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1280–1287.

Chuandong Su, Fumiyo Fukumoto, Xiaoxi Huang, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. DeepMet: A reading comprehension paradigm for token-level metaphor detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 30–39, Online. Association for Computational Linguistics.

Yuan Tian, Nan Xu, Wenji Mao, and Daniel Zeng. 2023. Modeling conceptual attribute likeness and domain inconsistency for metaphor detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7736–7752.

Xiaoyu Tong, Rochelle Choenni, Martha Lewis, and Ekaterina Shutova. 2024. Metaphor understanding challenge dataset for llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 3517–3536. Association for Computational Linguistics.

Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 680–690.

Lennart Wachowiak and Dagmar Gromann. 2023. Does GPT-3 grasp metaphors? identifying metaphor mappings with generative language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1018–1032. Association for Computational Linguistics.

Hai Wan, Jinxia Lin, Jianfeng Du, Dawei Shen, and Manrong Zhang. 2021. Enhancing metaphor detection by gloss-based interpretations. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1971–1981.

Shun Wang, Yucheng Li, Chenghua Lin, Loïc Barrault, and Frank Guerin. 2023. Metaphor detection with effective context denoising. *arXiv preprint arXiv:2302.05611*.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence*, 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial intelligence*, 11(3):197–223.

Yanzhi Xu, Yueying Hua, Shichen Li, and Zhongqing Wang. 2024. Exploring chain-of-thought for multimodal metaphor detection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 91–101.

Cheng Yang, Puli Chen, and Qingbao Huang. 2024. Can chatgpt's performance be improved on verb metaphor detection tasks? bootstrapping and combining tacit knowledge. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1016–1027. Association for Computational Linguistics.

Shenglong Zhang and Ying Liu. 2022. Metaphor detection via linguistics enhanced siamese network. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4149–4159.

A Experiment Details

All experiments with Llama7b were conducted on a single NVIDIA RTX 4090 GPU; all experiments with Llama13b were conducted on two NVIDIA RTX 4090 GPUs; and experiments with other large models were performed using APIs on a single NVIDIA RTX 4090 GPU.

A.1 Hyperparameters

(1) For the VUA ALL dataset, the learning rate was set to 3e-5, with a learning rate warmup, the number of iterations was 20, and the batch size was 32. (2) For the VUA Verb dataset, the learning rate was set to 3e-5, with a learning rate warmup, the number of iterations was 15, and the batch size was 32. (3) For the MOH-X dataset, the learning rate was set to 3e-5, with a learning rate warmup, the number of iterations was 15, and the batch size was 16. (4) For TroFi dataset, we use it only for zero-shot evaluation, where the model is trained on VUA ALL dataset and tested on TroFi dataset.

A.2 Average Running Time

The average processing time per data item is 3 seconds per iteration. The training and testing time for the VUA ALL dataset is around 100 hours, for the VUA Verb dataset it is around 30 hours, for the MOHX dataset it is around 0.5 hours, and for the TroFi dataset, the testing time is around 4 hours.