ParaMac: A General Unsupervised Paraphrase Generation Framework Leveraging Semantic Constraints and Diversifying Mechanisms

Jinxin Liu[1], **Jiaxin Shi**[3], **Ji Qi**[1], **Lei Hou**[1], **Juanzi Li**[1,2]*, **Qi Tian**[3]
[1] Department of Computer Science and Technology, BNRist;
[2] THU - Siemens Ltd., China Joint Research Center for Industrial Intelligence and IoT;
Tsinghua University, Beijing, China
[3] Huawei Cloud Computing Technologies Co., Ltd.
liujinxi20@mails.tsinghua.edu.cn

Abstract

Paraphrase generation reflects the ability to understand the meaning from the language surface form and rephrase it to other expressions. Recent paraphrase generation works have paid attention to unsupervised approaches based on Pre-trained Language Models (PLMs) to avoid heavy reliance on parallel data by utilizing PLMs' generation ability. However, the generated pairs of existing unsupervised methods are usually weak either in semantic equivalence or expression diversity. In this paper, we present a novel unsupervised paraphrase generation framework called Paraphrase Machine. By employing multi-aspect equivalence constraints and multi-granularity diversifying mechanisms, Paraphrase Machine is able to achieve good semantic equivalence and expressive diversity, producing a high-quality unsupervised paraphrase dataset. Based on this dataset, we train a general paraphrase model, which can be directly applied to rewrite the input sentence of various domains without any fine-tuning, and achieves substantial gains of 9.1% and 3.3% absolutely in BLEU score over previous SOTA on Quora and MSCOCO. By further fine-tuning our model with domain-specific training sets, the improvement can be increased to even 18.0% and 4.6%. Most importantly, by applying it to language understanding and generation tasks under the low-resource setting, we demonstrate that our model can serve as a universal data augmentor to boost the few-shot performance (*e.g.*, average 2.0% gain on GLUE). Code and data can be found at https://github.com/Matthewlliu/UnsupervisedParaphrase.

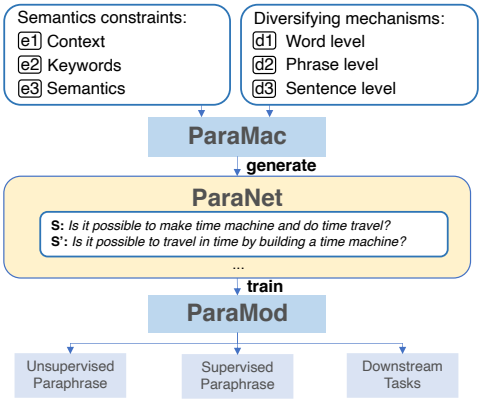
1 Introduction

Paraphrases are sentences that convey the same meaning with different forms of expressions. Automatic generation of paraphrases has been an essential task in natural language processing (NLP)

Input	Is it possible to make time machine and do time travel?
Output 1	Is it likely for time machine to do time travel?
Output 2	Is it possible to make time machine to time travel?
Ours	Is it possible to travel in time by building a time machine?

Inconsistency: affects **semantics** Repetition: affects **diversity**

(a) A example illustrating the problem of previous methods in terms of semantic equivalence and expression diversity.



(b) Our overall workflow.

Figure 1: We propose an unsupervised paraphrase generation framework named ParaMac. Based on that, a high-quality dataset named ParaNet is generated and used to train a general seq2seq pararphraser named ParaMod.

ever since the early days of the computational linguistics study (McKeown, 1979), and has a broad application on downstream tasks including question answering (Dong et al., 2017), semantic parsing (Berant and Liang, 2014; Wu et al., 2021), machine translation (Seraj et al., 2015), and etc. Additionally, paraphrase generation is a significant data augmentation method (Gao et al., 2020; Yu et al., 2020), which can benefit the learning in low-resource settings.

Early works like rule-based (McKeown, 1983; Barzilay and Lee, 2003) and thesaurus-based (Bolshakov and Gelbukh, 2004) methods generate paraphrases mainly by explicit manipulation on words, phrases, or sentences. But these methods usually perform poorly and are restricted by either heavy

* Corresponding author: lijuanzi@mail.tsinghua.edu.cn

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6193–6206
December 7-11, 2022

manual work or large language resources. Later, the sequence-to-sequence (Seq2Seq) paradigm is brought into the paraphrase generation (Prakash et al., 2016). By training on parallel annotations and combined with GAN (Yang et al., 2019) or VAE (Gupta et al., 2018), it greatly improves the performance. However, these supervised methods highly depend on large annotated parallel data, which is hard to acquire.

To overcome the difficulty in obtaining high-quality parallel corpora, recently, researchers begin to pay attention to unsupervised approaches using Pre-trained Language Models (PLMs) (Niu et al., 2021; Liu et al., 2020; Hegde and Patil, 2020; Meng et al., 2021), due to their great power in language modeling and understanding (Lin et al., 2019; Zhang et al., 2020). These existing works apply PLMs to paraphrase generation successfully and obtain good performances. However, they are still weak in either semantic equivalence or expression diversity, which are both necessary for a qualified rewriting. For example, methods that generate paraphrases by reconstructing or editing the original sentence (Liu et al., 2020; Hegde and Patil, 2020) usually only change common words locally, ignoring other global expression factors (*e.g.*, ordering) and thus hindering the diversity. On the other hand, methods that generate paraphrases from scratch (Meng et al., 2021; Niu et al., 2021) usually lack strong semantic constraints and thus suffer from an inevitable semantic divergence.

To tackle these problems, we propose a novel paraphrase generation framework called Paraphrase Machine (ParaMac), which leverages PLMs to generate paraphrases given an input and its context. For this framework, we propose multi-aspect equivalence constraints and multi-granularity diversifying mechanisms to produce various input expressions while keeping the original meaning as tight as it can. Specifically, we design the equivalence constraints from three aspects: the context, the keyword, and the overall semantics. On the other hand, we consider the diversifying mechanisms in three granularity: the word level, the phrase level, and the sentence level. All these constraints and mechanisms are combined to guarantee that the generated sentence preserves the semantics of the input with expressions as diverse as possible.

As shown in Figure 1, incorporated with these constraints and mechanisms, **ParaMac** can utilize PLMs' linguistic ability to generate unsupervised paraphrase pairs effectively. We generate a high-quality paraphrase dataset called Paraphrase Net (**ParaNet**) in an unsupervised way, which enable us to train a Paraphrase Model (**ParaMod**) based on a Seq2Seq PLM (*e.g.*, T5 (Raffel et al., 2020)).

By applying ParaMod directly to paraphrasing benchmarks (*i.e.*, Quora[1] and MSCOCO (Lin et al., 2014)) without any fine-tuning, we achieve a significant improvement over previous SOTA (*i.e.*, 9.1% and 3.3% absolutely in the BLEU score). After a further fine-tuning on domain-specific training data, ParaMod lifts the absolute improvements to even 18.0% and 4.6%. In addition, we want to highlight the framework's generality across downstream tasks, which is evaluated by applying ParaMod to language understanding and generation tasks to perform data generation or augmentation. We demonstrate that ParaMod is an excellent question generator that can cut down the manual work on question generation, and also a universal data augmentor to boost the performance of part of GLUE in the few-shot setting by an average of 2.0%.

2 Related Work

Supervised Approaches Typical supervised paraphrase generation methods mainly leverage annotated paraphrase pairs and neural Seq2Seq models (Prakash et al., 2016) such as LSTM (Hochreiter and Schmidhuber, 1997) or Transformer model (Vaswani et al., 2017). Following methods based on the Seq2Seq encoder-decoder architecture attempted to improve the performance by adding more constraints on generation. Bahdanau et al. (2015) tried to add attention and Cao et al. (2017); Gu et al. (2016) added copy mechanism to keep model focused on the important parts of input. VAE (Gupta et al., 2018) and GAN (Yang et al., 2019) enforced constraints from model and training aspects, respectively, to try to avoid unrealistic output. Some works leveraging other supervised signal can be categorized into zero-shot generation. For example, Mallinson et al. (2017); Wieting et al. (2017) made another attempt to generate paraphrase in a bilingual pivoting manner later known as back-translation. Guo et al. (2019) proposed to train a unified model on multilingual parallel data to achieve a one-step generation. Cai et al. (2021) further extended the pivoting idea from language to other semantic forms and explored the feasibility.

[1] https://www.kaggle.com/c/quora-question-pairs

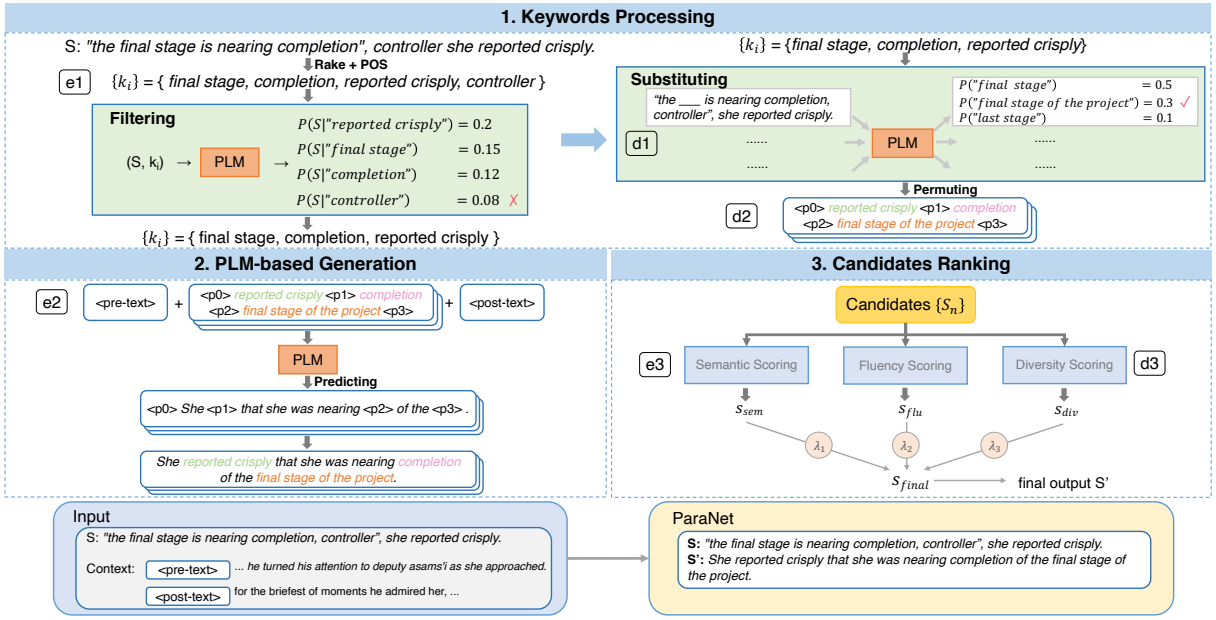


Figure 2: The complete generation process of Paraphrase Machine. The keywords filtering, substitution, PLM-based generation, and candidate evaluation steps are all achieved with PLMs in an unsupervised manner.

There also are some works (Iyyer et al., 2018; Sun et al., 2021) tried to incorporate syntactic structures to improve the diversity.

Supervised methods can usually get good performance, while the primary obstacle is getting large-scale and high-quality parallel data.

Unsupervised Approaches Unsupervised methods are hard to categorize since they are much less explored. Liu et al. (2020) transformed paraphrase generation into an optimization problem and utilized certain objectives to reflect the semantic equivalence and diversity. Siddique et al. (2020) shared a similar idea but optimized via deep reinforcement learning. Bowman et al. (2016) trained a VAE to reconstruct the input and sampled from the trained decoder to get its latent paraphrase. Roy and Grangier (2019) leveraged residual connections which allows a interpolation from classical auto-encoder to vector-quantized auto-encoder. Most recent works are focused on transformer-based PLMs. Meng et al. (2021) pre-trained a context-LM to generate paraphrase candidates with regularization of context. Other works directly used PLMs (*e.g.*, GPT-2 or BART) to generate paraphrase - Hegde and Patil (2020) used PLMs to reconstruct corrupted input, and Niu et al. (2021) brought up new blocking algorithm during generation to prevent PLMs from copying and repeating.

These methods intend to use PLMs in their process to improve performance. However, these generation results often suffer from either inconsistency of semantics or the lack of diversity.

3 ParaMac

In this section, we will introduce our unsupervised paraphrase generation framework ParaMac. Our idea starts from two basic assumptions: the first is the paraphrase must contain some key information of the original sentence, which we use keywords to represent, and they can be rendered in expressions; the second is there exists a different order for keywords to reform a fluency sentence, as long as proper connecting parts are filled between them.

Based on these two assumptions, our unsupervised paraphrase generation framework can be divided into three parts: 1) **Keywords Processing**: we extract the keywords of input, rephrase and reorder them; 2) **PLM-based Generation**: with the help of context regulation and the linguistic ability of PLMs, we connect the rephrased and reordered keywords into a fluency sentence as a paraphrase candidate; 3) **Candidates Ranking**: we use a series of metrics to select the best candidate.

To ensure semantic consistency during the process and the expression diversity of the outputs, we design multi-aspect equivalence constraints and multi-granularity diversifying mechanisms. Specifically, the equivalence constraints are:

- e1 Keywords constraint: keywords from the input are used as anchors in the output;

- (e2) Context constraint: we use context information to reduce the generation output space;
- (e3) Semantics constraint: we employ an automatic semantic evaluation on the candidates to get their sentence semantics ranks.

The diversifying mechanisms are embedded at three levels:

- (d1) Word level: keywords are replaced with their synonyms;
- (d2) Phrase level: the order of keywords is rearranged to change the possible collocation;
- (d3) Sentence level: a diversity score is used to encourage the different expressions.

Figure 2 provides an overview of the whole generation picture and where exactly these constraints and mechanisms are applied.

3.1 Keywords Processing

Keywords processing here consists of keywords extraction, filtering, substitution, and random permutation. In this paper, keywords refer to all the words and phrases extracted from the input.

Keywords Extraction & Filtering is to perform a coarse selection of the important information in the original sentence, which uses (e1). In this step, we leverage the Rake algorithm (Rose et al., 2010), which is an efficient keywords extraction algorithm based on word co-occurrence in libraries. It can return not only words but also phrases given the input sentence. To avoid missing important information, we also add the rest of the nouns and verbs of the input. Formally, given an input sentence S, we get its keywords set $\{k_i\}$.

Next, we intend to filter some of the redundant low informative keywords because the more keywords used in the later generation, the less diverse the output may be. Also, there will be a higher computational cost. So we want to relax the constraint by dropping certain a number of keywords.

In this paper, we measure the information of keyword k_i by computing $p(S|k_i)$, where the $p(\cdot|\cdot)$ represent the conditional generation score of PLMs. The intuition here is that if a keyword k_i is more likely to generate the whole sentence, the more informative and representative it is. According to the ranking of the score, we can filter out some of the low-scoring keywords

Keywords Substitution & Random Permutation aims to increase the diversity by rephrasing and reordering keywords, which uses (d1) and (d2).

The rephrasing, namely the substitution of synonym, is achieved by masking the target keyword and utilizing masked language models (MLMs) such as BERT or T5 to predict the masked token. As shown in the substituting step in Figure 2, the second prediction of beam search is regarded as a semantically equivalent substitution. Note that although all the keywords will go through this operation, sometimes the keyword stays the same if we use T5 as the MLM. Additionally, some handcrafted blocking rules based on WordNet are also used to avoid the situation that MLMs replace a word with an opposite meaning (*e.g.*, *large sofa* to *small sofa*, both of which can sometimes fit in the context).

After the substitution, keywords are randomly re-shuffled into many different orders, filled with span-mask tokens to form the final input to the PLM.

3.2 PLM-based Generation

We use the powerful PLMs to generate fluency and meaningful sentences. Meanwhile, we use (e1) and (e2) to reduce the possible output space and prevent an overly free generation because they make the generated output have to fit in the original context and contain keywords of the input.

Specifically, we choose bidirectional model *T5-large* as our PLM and leverage the pre-training task of T5 described in (Raffel et al., 2020). As shown in the PLM-based generation step of Figure 2, the input to T5 is formalized by connecting the keywords with span-masking tokens, concatenated with the context of the original sentence S. Then, the PLM is used to predict the masked spans, and we fill the predictions back to the input. In this way, we ensure that the key information of S is kept in output, and the model is context-aware during the generation.

After the generation, the generated outputs will be evaluated as candidates in the next step.

3.3 Candidates Ranking

In this section, we describe the multiple scoring functions applied in the evaluation of the quality of candidates. We consider the quality of a sentence from three aspects: semantic equivalence, fluency, and diversity.

3.3.1 Semantic Score

Semantic score s_{sem} is designed according to the third semantic constraint (e3). Due to the scale of data we deal with, we use the Bert-Score (Zhang et al., 2020) as an automatic evaluation method. It is based on the embedding of the tokens to measure the semantic similarity of a pair of sentences.

3.3.2 Fluency Score

The fluency of generated sentences should also be highly valued, and here we choose the candidate's perplexity score to reflect the fluency. However, since the perplexity is the smaller the better, in practice we calculate the probability of the sentence as the metric s_{flu}. Specifically, we use a PLM to calculate s_{flu} using its next-word prediction probability.

$$\begin{aligned} s_{flu}(S_1, S_2) &= P(w_1) \cdot P(w_2|w_1) \\ &\cdot P(w_3|w_1w_2) \cdot \cdots \cdot P(w_n|w_1 \dots w_{n-1}) \end{aligned} \tag{1}$$

3.3.3 Diversity Score

This measurement is a direct expression of (d3), where we intend to encourage the diversity of the generated sentences. Specifically, both wording and words' order are considered in this case. Inspired by Ment et al (Meng et al., 2021), we use Jaccard distance to measure the difference of two sentences S_1 and S_2. The score s_{div} can be calculated by the following equation:

$$\begin{aligned} s_{div}(S_1, S_2) &= \beta_1 \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} + \\ \beta_2 &\frac{1}{|S_1 \cap S_2|} \Sigma_{w \in S_1 \cap S_2} \frac{|p_{S_1}(w) - p_{S_2}(w)|}{max(|S_1|, |S_2|)} \end{aligned} \tag{2}$$

where S is considered as a set of its words w, and $p_S(w)$ means the position of word w in S.

3.3.4 Comprehensive Score

Eventually, all the three evaluation scores are taken into consideration. We leverage the linear combination of s_{sem}, s_{flu}, and s_{div} to calculate the final comprehensive score s_{final} as follows:

$$\begin{aligned} s_{final}(S_1, S_2) &= \lambda_1 \cdot s_{sem}(S_1, S_2) + \\ &\lambda_2 \cdot s_{flu}(S_1, S_2) + \lambda_3 \cdot s_{div}(S_1, S_2) \end{aligned} \tag{3}$$

where λ_1, λ_2, and λ_3 are weight parameters.

4 ParaNet and ParaMod

With the above framework, we can now choose a proper corpus to generate our paraphrase dataset ParaNet. As the generation process needs the context of inputs, we choose the long-form BookCorpus (Zhu et al., 2015) as our generation input corpus. Since T5 uses BookCorpus as one of its pre-training corpora and we don't want the PLM of ParaMac to have seen the input sentences before, we use a newly crawled version[2] (Sep. 2020 by Shawn Presser) excluding the original BookCorpus, which finally leaves us 3551 books. The genres of these books include fiction, nonfiction, essay, poetry, plays, and screenplays, ranging from up to 100 topics such as romance, science fiction, fantasy, thriller, and suspense.

From this subset of BookCorpus, we randomly sample 10k examples. Each input examples contains 1) a complete sentence S with a length between 60 and 100 characters; 2) the context before and behind S, both with an average length of 250 characters. Given the 10k examples, we generate the ParaNet in an unsupervised way using ParaMac. Then, based on ParaNet, we are able to train a Seq2Seq language model ParaMod, which can generate paraphrased sentences given any sentence. The implementation details can be found in Appendix A.

5 Experiments

In this section, we evaluate our proposed paraphrase generation model ParaMod in both unsupervised and supervised settings. Furthermore, we use our ParaMod as a data augmentation and generation tool to validate its effectiveness in downstream NLP tasks.

5.1 Paraphrase Generation

ParaMod can be directly applied to different domains of datasets without further fine-tuning, so we consider this setting unsupervised. The supervised setting refers to further fine-tuning ParaMod on domain-specific data.

To demonstrate the generality of our model, we choose Quora and MSCOCO as the evaluation datasets, which represent interrogative and declarative sentences, respectively.

Quora[3] dataset is also popularly known as the Quora question pair. The latest version contains 149k parallel paraphrase question pairs and 260k non-parallel questions and we follow the split used

[2] https://github.com/soskek/bookcorpus

[3] https://www.kaggle.com/c/quora-question-pairs

in Liu et al. (2020); Meng et al. (2021).
MSCOCO (Lin et al., 2014) is originally used for image caption, consisting of roughly 120k images, annotated with five captions each. We follow the standard split (Lin et al., 2014).

5.1.1 Baselines and Metrics

For the unsupervised setting, we compare our model to the following works as our baselines.
VAE Bowman et al. (2016) generated paraphrases by sampling from a continuous space.
CGMH Miao et al. (2019) utilized Metropolis-Hasting sampling strategy for sentence generation.
UPSA Liu et al. (2020) introduced a novel approach that transform the paraphrase generation into a optimization problem.
DB Niu et al. (2021) brought up a blocking mechanism to diversify the output of PLMs.
CorruptLM Hegde and Patil (2020) utilized GPT-2 to generate paraphrase by teaching PLMs to recover the input from corrupted sentence.
ConRPG Meng et al. (2021) proposed to generate paraphrase by using the context as the regularizer.

For the supervised setting, the baselines are:
ResLSTM Prakash et al. (2016) trained a stacked residual LSTM to do Seq2Seq paraphrasing.
VAE-SVG-eq Gupta et al. (2018) combined VAE and LSTM to generate realistic paraphrase.
Transformer Vaswani et al. (2017) developed the Transformer with attention mechanism.
DNPG Li et al. (2019) designed a multi-granularity Transformer-based model.
ConRPG The unsupervised model of Meng et al. (2021) can be further trained on supervised data.
LBoW Fu et al. (2019) used a discrete bag-of-words as the latent encoding for the encoder-decoder generation model.
SCSVED Chen et al. (2020) leveraged adversarial learning on variational encoder-decoder to help keep semantics consistent.

We copy the results of these methods according to their papers if they have done the same experiment under the same settings.

The metrics used are iBLEU score (Sun and Zhou, 2012), BLEU score (Papineni et al., 2002), and ROUGE score (Lin, 2004). The BLEU and ROUGE score are the most widely-used metrics for sentence similarity. The iBLEU extends BLEU by penalizing the similarity between the generated sentence and the input, in which we adopted the same weight parameter as Meng et al. (2021). Finally, as with previous works (Chen et al., 2020; Gupta et al., 2018), we compute the value of these metrics by generating multiple paraphrases and select the best one with the highest iBLEU score.

5.1.2 Unsupervised Results

The main results of unsupervised paraphrase generation are in Table 1. It shows that the proposed ParaMod outperforms baselines on every metric. In particular, there is a huge lift in BLEU on Quora. On MSCOCO, there is a significant increase in Rouge values and an approximate 3.3% improvement in BLEU. The iBLEU score is also comparable with previous unsupervised methods. This result shows the potential and value of our ParaNet. One advantage of our model is that we smoothly utilize the original pre-training task in the generation process, which has no gap with the PLMs' pre-training.

Model	Quora			
	iBLEU	BLEU	R1	R2
VAE	8.16	13.96	44.55	22.64
CGMH	9.94	15.73	48.73	26.12
DB	9.60	14.10	59.90	28.50
UPSA	12.02	18.18	56.51	30.69
CorruptLM	12.32	17.97	59.14	32.14
ConRPG	12.68	18.31	59.62	33.10
ParaMod	**14.57**	**27.45**	**60.05**	**39.32**
Model	MSCOCO			
	iBLEU	BLEU	R1	R2
VAE	7.48	11.09	31.78	8.66
CGMH	7.84	11.45	32.19	8.67
UPSA	9.26	14.16	37.18	11.21
CorruptLM	10.32	15.60	38.12	12.40
ConRPG	11.17	16.98	39.42	13.50
ParaMod	**11.34**	**20.31**	**52.93**	**29.11**

Table 1: Unsupervised paraphrase generation results. The baseline figures are copied from Meng et al. (2021) and Niu et al. (2021)

5.1.3 Supervised Results

To demonstrate the strong power and generality of ParaMod, we only randomly select subsets of the whole training set. Specifically, we sample a 500 and a 10k training set on Quora and MSCOCO, respectively, then fine-tune ParaMod on these subsets for two epochs. The results of supervised paraphrase generation are shown in Table 2. On Quora, merely a 500-example 2-epoch fine-tuning can enable ParaMod to outperform all baselines, and a 10k-example 2-epoch fine-tuning dramatically improves the performance in all metric val-

ues. On MSCOCO, the 500-example fine-tuning makes ParaMod comparable to previous SOTA, and a further 10k-example fine-tuning outperforms it with significant enhancement. This result shows the effectiveness of our ParaMod as a general initialization for paraphrase models, and also further validates the value of ParaNet.

Model	Quora			
	iBLEU	BLEU	R1	R2
ResLSTM	12.67	17.57	59.22	32.40
VAE-SVG-eq	15.17	20.04	59.98	33.30
Transformer	16.25	21.73	60.25	33.45
DNPG	18.01	25.03	63.73	37.75
ConRPG	19.96	26.81	65.03	38.49
SCSVED	-	26.04	60.28	35.26
ParaMod^{500}	20.21	36.26	67.23	47.50
ParaMod^{10k}	**28.36**	**44.77**	**73.25**	**55.92**
Model	MSCOCO			
	iBLEU	BLEU	R1	R2
ResLSTM	11.04	21.65	40.11	14.31
ResLSTM-att	11.59	23.66	41.07	15.26
VAE-SVG-eq	14.13	25.99	40.10	15.18
LBoW	14.02	25.27	42.08	16.13
SCSVED	-	27.33	40.65	15.39
ParaMod^{500}	14.32	27.95	63.58	38.59
ParaMod^{10k}	**19.0**	**31.93**	**66.2**	**42.27**

Table 2: Supervised paraphrase generation results. Baseline figures on Quora are mainly referred from Meng et al. (2021), figures on MSCOCO are mainly referred from Zhou and Bhat (2021). ^{500} and ^{10k} stands for the model is fine-tuned 500 and 10k subsets, repectively.

5.1.4 Case Study

Here we provide some real examples generated by unsupervised models on Quora. Although ConRPG is the best baseline model, its model and code are both unavailable. Therefore we choose the second-best CorruptLM as a comparison to our ParaMod. The examples are shown in Table 3. It can be seen that the output of ParaMod is more fluency, diverse, and accurate in semantics.

5.2 Downstream Tasks

In this work, we particularly highlight our model's generality across different downstream tasks, especially in low-resource settings. We consider two situations - the first application scenario is the question generation in tasks like semantic parsing and question answering; the second application is the few-shot learning for NLP tasks.

5.2.1 Low-Resource Generation

In this experiment, we consider the application of Knowledge-based Question Answering (KBQA), which aims to answer the given natural language question based on the knowledge base. Recently, one prominent approach to constructing datasets for KBQA is the synthesizing-then-paraphrasing pipeline (Lan et al., 2021). First, template questions are generated automatically, and then crowd-sourced workers are recruited to paraphrase the template questions into the natural ones (Wang et al., 2015; Gu et al., 2021; Cao et al., 2022). Although this two-stage paradigm makes constructing large-scale datasets possible, the time and human efforts for paraphrasing are intensive and costly.

Model	KQA Pro			
	iBLEU	BLEU	R1	R2
T5-base^{100}	6.61	17.13	41.91	28.86
T5-base^{500}	8.69	20.46	44.18	31.32
T5-base^{1k}	17.31	34.62	63.26	46.23
ParaMod^{100}	16.67	31.33	66.72	46.02
ParaMod^{500}	17.44	33.75	68.31	47.82
ParaMod^{1k}	19.52	36.54	69.98	49.91

Table 4: Results for paraphrasing template question to natural language question on KQA Pro.

We aim to validate the effectiveness of ParaMod to paraphrase questions for KBQA automatically. We adopt the dataset KQA Pro (Cao et al., 2022) since it is a large-scale Complex KBQA dataset whose template questions are generated according to a synchronous grammar and then paraphrased

Input	CorruptLM	ParaMod (ours)
Does Lipton green tea assist in weight loss?	Does green tea assist for weight loss?	Is lipton green tea an aid in weight loss?
Can you create another upwork account after suspension?	how do i add another upwork account to suspension?	After the suspension, can you open another upwork account?
Why is it that the American government is so corrupt?	Are the government corrupt?	Why is the American government so corrupt?
Are there any verified angel investors on quora?	What angel investors are on quora?	Does quora have any verified angel investors?

Table 3: Examples of the generation output of Our method and CorruptLM (Hegde and Patil, 2020) on Quora.

by AMT workers. To validate the low-resource generation ability, we compare paraphrase models in the few-shot setting. We split 10k question pairs as the test set, leaving the rest for training use. The models are fine-tuned on subsets of the training set, which contain 100, 500, and 1k randomly selected examples, respectively. We compare our ParaMod with *T5-base*. The second-best unsupervised model CorruptLM is not used because it is not a general model and needs to train on domain-specific text, which we don't have in a low-resource setting. We use the same metrics as the paraphrase generation tasks in the last two sections. The results are demonstrated in Table 4. It is clear that ParaMod's performance greatly exceeds *T5-base*. Especially when there are only very few golden human-written pairs, ParaMod is still able to achieve reasonably good generation quality.

5.2.2 Low-Resource Augmentation

In this experiment, we intend to demonstrate that ParaMod is also a universal data augmentor. We follow Gao et al. (2021), who developed a prompt-tuning method for the few-shot application on GLUE. We follow their experiment setting, choosing part of the GLUE tasks, and expand their K-shot data with our ParaMod. The baselines here are two other data augmentation methods, including the naive inserting/deleting/replacing by PLMs and the CorruptLM (trained on MSCOCO).

From the results shown in Table 5, we see that our ParaMod improves the performance of the chosen tasks by an average of 2.0% and also improves the stability of all tasks (smaller std values). It also shows that the naive augmentation and CorruptLM augmentation both cause a drop in performance. The possible reason is that the augmented examples might be semantically inconsistent with the origin example, and the few-shot model is very sensitive to out-of-distribution examples. It is also worth noting that we observe the performance drop when the augmenting size N increases for all augmentation methods. We consider this a normal phenomenon, for the extra training examples we created here are all similar to the original ones in terms of semantics, and training on too many similar examples can cause over-fitting.

6 Ablation Studies

In this section, we study various factors that can affect the performance of our ParaMod.

6.1 Paraphrase Pre-Training Data Size

We intend to explore the influence of the size of ParaNet used to train ParaMod. In Table 6, we train *T5-base* for 3 epochs on subsets of ParaNet that contains 25k, 50k, 75k pairs respectively, and evaluated the model on Quora. We can see from the results that the performance of paraphrasing on Quora gradually rises when we increase the amount of training data, showing that our ParaNet is helpful for paraphrase generation.

Size	Quora			
	iBLEU	BLEU	R1	R2
0	8.72	16.23	50.43	31.43
25k	13.30	25.76	57.47	37.57
50k	13.41	26.01	57.78	37.84
75k	13.62	26.57	58.77	38.50
100k	**14.57**	**27.45**	**60.05**	**39.32**

Table 6: The influence of training data size on ParaMod's performance.

6.2 Paraphrase Pre-Training Epochs

We train ParaMod on 10k ParaNet pairs on Quora with different epochs. From Table 7 we can observe that with the increase of the training epochs, the performance improves significantly at first, but then gradually declines, suggesting a slight over-fitting during training.

Model (metric)	SST-2 (acc)	SST-5 (acc)	MNLI (acc)	MNLI-mm (acc)	SNLI (acc)	MRPC (f1)
Prompt-based FT	93.1(0.3)	49.5(1.7)	70.0(3.6)	72.0(3.1)	77.5(3.5)	76.7(5.7)
Fine-tuning (full)	95.0	58.7	89.8	89.5	92.6	91.4
Naive$_{N=3}$	93.2(0.6)	47.6(2.4)	67.6(1.5)	69.1(1.5)	75.0(4.8)	76.9(4.1)
CorruptLM$_{N=3}$	92.9(0.7)	47.8(4.3)	63.7(2.8)	65.4(2.4)	75.3(2.4)	74.5(2.3)
ParaMod$_{N=3}$	**93.7(0.1)**	**51.7(0.7)**	**72.5(2.2)**	**73.9(1.8)**	**80.9(1.6)**	**77.9(3.9)**

Table 5: The results of few-shot learning on GLUE. We report mean (and standard deviation) performance over 5 different splits (Gao et al., 2021). $N = 3$ means we increase the origin K-shot training set size to its N times. The CorruptLM used in this experiment is trained on MSCOCO.

Epochs	Quora			
	iBLEU	BLEU	R1	R2
0	8.72	16.23	50.43	31.43
5	**14.57**	**27.45**	**60.05**	**39.32**
10	12.59	26.23	58.79	38.16
15	12.05	26.22	59.09	38.11
20	11.34	26.08	59.62	38.27
25	11.07	26.02	59.73	38.23

Table 7: The influence of training epochs on ParaMod's performance.

7 Conclusions

In this paper, we introduce a novel unsupervised paraphrase generation framework ParaMac, which utilizes PLMs' linguistic ability, combined with multi-aspect equivalence constraints and multi-granularity diversifying mechanisms, to improve the generation quality in terms of semantic equivalence and expression diversity. Moreover, we demonstrate the generality and value of our general paraphrase model in several downstream tasks.

Limitations

In this section, we intend to point out the three limitations of our unsupervised paraphrase generation framework. The first is that this framework requires a relatively high-quality corpus, since the paraphrase generation needs to use the context of inputs; Secondly, Our framework is relatively complicated. Due to the limitations of the input data, ParaMac cannot directly perform generation in paraphrase generation tasks such as Quora. In order to generate a paraphrase given any sentence, we produce ParaNet and ParaMod to do the job; Finally, The construction of ParaNet is demanding on computing resources. In practice, it needs about 30-50s to produce a pair on a 24GB RTX 3090, depending on the hyper-parameters of implementation.

8 Acknowledgments

We would like to thank all the anonymous reviewers for their careful work and thoughtful suggestions. This work is supported by the New Generation Artificial Intelligence of China (2020AAA0106501), grants from the Institute for Guo Qiang, Tsinghua University (2019GQB0003) and Tsinghua University-Siemens Ltd., China Joint Research Center for Industrial Intelligence and Internet of Things. This work is also supported by Huawei Cloud Computing Technologies Co., Ltd.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 1415–1425. The Association for Computer Linguistics.

Igor A. Bolshakov and Alexander F. Gelbukh. 2004. Synonymous paraphrasing using wordnet and internet. In *Natural Language Processing and Information Systems, 9th International Conference on Applications of Natural Languages to Information Systems, NLDB 2004, Salford, UK, June 23-25, 2004, Proceedings*, volume 3136 of *Lecture Notes in Computer Science*, pages 312–323. Springer.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21. ACL.

Yitao Cai, Yue Cao, and Xiaojun Wan. 2021. Revisiting pivot-based paraphrase generation: Language is not the only optional pivot. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4255–4268.

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. 2022. KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 6101–6119. Association for Computational Linguistics.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3152–3158. AAAI Press.

Wenqing Chen, Jidong Tian, Liqiang Xiao, Hao He, and Yaohui Jin. 2020. A semantically consistent and syntactically variational encoder-decoder framework for paraphrase generation. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1186–1198. International Committee on Computational Linguistics.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 875–886. Association for Computational Linguistics.

Yao Fu, Yansong Feng, and John P. Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13623–13634.

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Paraphrase augmented task-oriented dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 639–649. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Yu Gu, Sue E. Kase, Michelle T. Vanni, Brian M. Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond i.i.d.: Three levels of generalization for question answering on knowledge bases. *Proceedings of the Web Conference 2021*.

Yinpeng Guo, Yi Liao, Xin Jiang, Qing Zhang, Yibo Zhang, and Qun Liu. 2019. Zero-shot paraphrase generation with multilingual language models. *CoRR*, abs/1911.03597.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5149–5156. AAAI Press.

Chaitra V. Hegde and Shrikumar Patil. 2020. Unsupervised paraphrase generation using pre-trained language models. *CoRR*, abs/2006.05477.

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

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1875–1885. Association for Computational Linguistics.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *IJCAI*.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3403–3414. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*, pages 241–253. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 302–312. Association for Computational Linguistics.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893.

Kathleen R. McKeown. 1979. Paraphrasing using given and new information in a question-answer system. In *17th Annual Meeting of the Association for Computational Linguistics, 29 June - 1 July 1979, University of California at San Diego, La Jolla, CA, USA*. ACL.

Kathleen R. McKeown. 1983. Paraphrasing questions using given and new information. *Am. J. Comput. Linguistics*, 9(1):1–10.

Yuxian Meng, Xiang Ao, Qing He, Xiaofei Sun, Qinghong Han, Fei Wu, Chun Fan, and Jiwei Li. 2021. Conrpg: Paraphrase generation using contexts as regularizer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 2551–2562. Association for Computational Linguistics.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: constrained sentence generation by metropolis-hastings sampling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6834–6842. AAAI Press.

Tong Niu, Semih Yavuz, Yingbo Zhou, Nitish Shirish Keskar, Huan Wang, and Caiming Xiong. 2021. Unsupervised paraphrasing with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5136–5150. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek V. Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2923–2934. ACL.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*, pages 1 – 20.

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6033–6039. Association for Computational Linguistics.

Ramtin Mehdizadeh Seraj, Maryam Siahbani, and Anoop Sarkar. 2015. Improving statistical machine translation with a multilingual paraphrase database. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1379–1390. The Association for Computational Linguistics.

A. B. Siddique, Samet Oymak, and Vagelis Hristidis. 2020. Unsupervised paraphrasing via deep reinforcement learning. *CoRR*, abs/2007.02244.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual SMT system for paraphrase generation. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*, pages 38–42. The Association for Computer Linguistics.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. 2021. AESOP: paraphrase generation with adaptive syntactic control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5176–5189. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *ACL*.

John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. *arXiv preprint arXiv:1706.01847*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers), pages 1112–1122. Association for Computational Linguistics.

Shan Wu, Bo Chen, Chunlei Xin, Xianpei Han, Le Sun, Weipeng Zhang, Jiansong Chen, Fan Yang, and Xunliang Cai. 2021. From paraphrasing to semantic parsing: Unsupervised semantic parsing via synchronous semantic decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5110–5121. Association for Computational Linguistics.

Qian Yang, Zhouyuan Huo, Dinghan Shen, Yong Cheng, Wenlin Wang, Guoyin Wang, and Lawrence Carin. 2019. An end-to-end generative architecture for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3130–3140. Association for Computational Linguistics.

Junjie Yu, Tong Zhu, Wenliang Chen, Wei Zhang, and Min Zhang. 2020. Improving relation extraction with relational paraphrase sentences. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1687–1698. International Committee on Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5075–5086. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

A Implementations

In practice, our ParaMac works with a keywords filtering rate of $round(|\{k_i\}| \cdot 0.15)$, where $\{k_i\}$ is the set of keywords; The number of keyword permutations is set to a maximum of 15; the PLM we use in fluency score computation is GPT-2;the base language model of Bert-Score is a Roberta-Large specially fine-tuned on the MNLI (Williams et al., 2018) task to increase the accuracy; the weighting parameters in the comprehensive score are set to $\lambda_1 = 4.0, \lambda_2 = 8.0$, and $\lambda_3 = 1.2$. These figures are set manually by experiments - we take 500 examples generated by ParaMac and calculate each score's standard deviation (std). To avoid one score totally overwhelming the other, we set the weights inversely proportional to its corresponding score's std.

For ParaMod, we choose the *T5-base* as the base model, using the 10k-pair ParaNet as the training set, training with AdamW optimizer, learning rate of 1e-4, and $\beta_1 = 0.9, \beta_2 = 0.999$. Due to the limitation of GPU memory, we set the batch size to 4 and gradient accumulation steps to 4 as well. The model converges after training about 20 hours for 25 epochs on a single 24GB RTX 3090, while we observe by experiment that five epochs of training yield the best performance.

All the experiments are completed by Pytorch and the transformers toolkit. We use the transformers' *Trainer* class to build the Seq2Seq model code base.

B Error Analysis

In this section, we want to provide an error analysis on some bad cases in our experiment. Before the large-scale generation of ParaNet, we conducted a small-batch experiment, and revised some of the error modes by human observation. The observed errors can be categorized into four types:

- The confusion of affirmative and negative. The model often ignores the negative suffix in *aren't/isn't/haven't*, and output are/is/have. This is mitigated by adding *n't* as a special keyword if it's in the sentence.
- The confusion of antonyms. In keyword substitution, the model sometimes fills in the antonym, e.g., large to small. This is prevented by using the word's synonym set in WordNet.
- Missing important part. This is mainly caused by the incomplete keyword extraction of RAKE. To alleviate this problem, we consider all the nouns and verbs as extra keywords and lower the filtering rate to avoid dropping important information.
- Punctuation symbols problems. Rake isn't good at handling punctuation symbols. Symbols are sometimes included in a keyword, thus restricting its output position. Also, Rake splits words connected with·-". For example, the keyword "semi-fluidic nature" will be split into "semi" and "fluidic nature", which greatly harms the semantics after reordering. We add extra rules to avoid these problems.

Examples of these error types are listed accordingly in Table 8. The first example changes the original negative saying to affirmative; the second generates *small* rather than the synonyms of *large* in the input; the third one misses *station* in its keywords, thus generates the output with different semantics; The last one fails to keep consistent with the input because the keyword *semi-fluidic* is splited.

Input	Output
Such a possibility hadn't even been discussed during the planning stages.	This was discussed during the planning stages as a possibility.
A large sofa was shoved against the wall, covered in a thin blanket.	A small sofa was wrapped in a soft blanket and tucked against the wall.
"As you command, controller," grudy said, and returned to his station.	"I returned to the controller," grudy said.
"Does the semi-fluidic nature of the crystals present a weakness in that regard.?"	"Is there any weakness in the semi-crystalline structure in that regard?" the fluidic nature present.

Table 8: Examples of the error occurred in the generation of ParaNet.

Although error modes were revised and avoided when spotted in this optimizing process, there still exists some semantic errors in the final ParaNet we used. Nevertheless, these errors do not affect the overall quality of the dataset much. In terms of an automatically generated parallel dataset, the quality of ParaNet can still be said to be very good, which has also been demonstrated by our experimental results.