Multi-task Learning for Paraphrase Generation With Keyword and Part-of-Speech Reconstruction

Xuhang Xie **Xuesong Lu*** **Bei Chen**
School of Data Science and Engineering
East China Normal University
{xhxie@stu,xslu@dase,beichen@stu}.ecnu.edu.cn

Abstract

Paraphrase generation using deep learning has been a research hotspot of natural language processing in the past few years. While previous studies tackle the problem from different aspects, the essence of paraphrase generation is to retain the key semantics of the source sentence and rewrite the rest of the content. Inspired by this observation, we propose a novel two-stage model, **PGKPR**, for **p**araphrase **g**eneration with **k**eyword and **p**art-of-speech **r**econstruction. The rationale is to capture simultaneously the possible keywords of a source sentence and the relations between them to facilitate the rewriting. In the first stage, we identify the possible keywords using a prediction attribution technique, where the words obtaining higher attribution scores are more likely to be the keywords. In the second stage, we train a transformer-based model via multi-task learning for paraphrase generation. The novel learning task is the reconstruction of the keywords and part-of-speech tags, respectively, from a perturbed sequence of the source sentence. The learned encodings are then decoded to generate the paraphrase. We conduct the experiments on two commonly-used datasets, and demonstrate the superior performance of PGKPR over comparative models on multiple evaluation metrics.

1 Introduction

The task of paraphrase generation is to rephrase a given sentence by preserving its key semantics. While the problem was solved using rule-based approaches (McKeown, 1979; Meteer and Shaked, 1988) and traditional machine learning techniques (Quirk et al., 2004; Wubben et al., 2010), recent attentions have been shifted to devising effective deep neural networks (Prakash et al., 2016; Gupta et al., 2018; Li et al., 2018), which generally adopt the encoder-decoder framework. More recently, controllable paraphrase generation has been extensively investigated and offers the mechanisms to guide the generation process by providing a reference such as a syntactic template (Iyyer et al., 2018; Goyal and Durrett, 2020; Huang and Chang, 2021), a sentential exemplar (Chen et al., 2019; Su et al., 2021) and so on.

SRC	what are good workouts to lose belly fat ?
POS	[WDT] [VBP] [TO] [.]
TGT	what is the best way to lose belly fat ?
POS	[WP] [VBZ] [DT] [TO] [.]
GNT	what are some good exercises to get rid of belly fat ?
POS	[WDT] [VBP] [DT] [TO] [.]

Table 1: A running example.

Although the problem has been studied from different aspects, the fundamental goal of paraphrase generation is to preserve the key semantics of a source sentence and rewrite the rest of the content. Taking the paraphrase pair in Table 1 as a running example, the key semantics are entailed by the words "good workouts", "lose belly fat" and "best way" in the source (SRC) and target (TGT) sentence, respectively. The rest of the content can be considered as auxiliary words that express the relations between the keywords. Inspired by the observation, we propose to enhance the representativeness of the encodings of a source sentence by learning simultaneously the possible keywords and the relations between them, before the encodings are fed into the decoder for text generation. To this end, we use a prediction attribution technique (Li et al., 2016a) to identify the possible keywords and use the part-of-speech (POS) tags to label the rest of the words, which represent the relations between the keywords. Table 1 shows the predicted keywords (in red) and the POS tags of the other words in the source sentence. Finally, the sentence

* Xuesong Lu is the corresponding author.

Findings of the Association for Computational Linguistics: ACL 2022, pages 1234 - 1243

generated (GNT) by our model successfully preserves both the semantics of the keywords using synonyms (in blue) and the relations between the keywords using the auxiliary words with similar POS tags.

Specifically, we propose a novel two-stage model, **PGKPR**, for **p**araphrase **g**eneration with **k**eyword and **p**art-of-speech **r**econstruction. In the first stage (Section 3), we fine-tune a BERT model to identify the keywords in a source sentence. The identification is based on a prediction attribution technique (Li et al., 2016a) that computes the gradient vector of each input word. We compute as the attribution score of each input word the L2-norm of the corresponding gradient vector, where the words with higher scores are more likely to be the keywords. In the second stage(Section 4), we adopt Transformer (Vaswani et al., 2017) and devise a multi-task learning model for paraphrase generation. Given a pair of paraphrase sentences, the learning tasks include 1) reconstructing the keywords and the POS tags of all words in the source sentence, 2) distinguishing the latent features of the pair from the features of non-paraphrase pairs, and 3) generating the paraphrase sentence. Finally, the objective function is the combination of the loss function in each learning task. In the experiments, we show that PGKPR outperforms the comparative models by a notable margin on both BLEU and ROUGE scores. The ablation study shows the effectiveness of each learning task, and the case study and user study show that PGKPR could produce paraphrases with higher quality.

A similar study was conducted by (Su et al., 2021), where they proposed a novel identification algorithm, PSI, to identify the primary and secondary content in a source sentence. Our work differs from theirs at least on the following three aspects. First, our strategy for keyword identification is purely data-driven, whereas the PSI algorithm uses a rule-based method and is sensitive to the similarity measurement used in the algorithm. Second, the PGKPR model is trained with multiple learning tasks, whereas the IANet model proposed in (Su et al., 2021) only has the learning task of predicting the target sentence. Third, PGKPR determines the keywords in a source sentence with the probability transformed from the attribution scores, which gives the model a more flexible way to separate the keywords and the other content, whereas the IANet model deterministically separates the primary and secondary content using a manually-tuned threshold based on the PSI scores.

2 Related Work

2.1 Paraphrase Generation

Recent studies have extensively applied various deep learning techniques for paraphrase generation. Representative studies have devised stacked residual LSTM networks (Prakash et al., 2016), copy mechanisms (Cao et al., 2017), reinforcement learning algorithms (Li et al., 2018), and unsupervised training methods (Roy and Grangier, 2019), etc. While performing much better than rule-based methods, these models do not offer user-defined mechanisms to control the paraphrase generation process. As such, (Iyyer et al., 2018) propose to generate paraphrases conditioned on a user-provided syntax template. (Chen et al., 2019) propose to extract the syntax exemplar from a given sentence instead of using a syntax template. (Goyal and Durrett, 2020) propose to perturb the preorder of the syntax structure of a source sentence for paraphrase generation. Two studies are related to our work. (Su et al., 2021) propose a Primary/Secondary Identification algorithm to separate the primary and secondary content of a source sentence. (Fu et al., 2019) propose to sample a latent bag of words from the encoder, which is an implicit way of extracting the keywords of a source sentence.

2.2 Prediction Attribution Techniques

Given a trained model, a prediction attribution technique calculates the attribution (i.e., contribution) of each input unit to a model prediction, which explains the faithfulness or reasoning process of the model (Bastings and Filippova, 2020). Representative techniques include gradient-based methods (Baehrens et al., 2010; Li et al., 2016a; Sundararajan et al., 2017), propagation-based methods (Bach et al., 2015; Arras et al., 2017; Binder et al., 2016) and occlusion-based methods (Zeiler and Fergus, 2014; Li et al., 2016b). The method used in the current work is the first-derivative saliency (i.e., the gradient) (Li et al., 2016a), which belongs to the first category. Take NLP models for example, an input unit in NLP tasks is usually the embedding of a word. Given a model's output, the method computes the gradient vector of the output with respect to the input embedding, and takes the L2-norm of the gradient vector as the contribution of the input to the output.

3 Stage One: Keyword Prediction

In the first stage, we train a BERT model to predict the keywords in a source sentence. The prediction is based on an attribution technique that computes the gradients of the input elements (Li et al., 2016a). In particular, given a binary classification model f and an input sequence $\mathcal{X} = \{x_1, x_2, \ldots, x_l\}$, where l is the number of input elements (i.e., the sequence length), the gradient vector g_i of x_i $(1 \leq i \leq l)$ is computed as,

$$g_i = \nabla_{x_i} f(\mathcal{X}), \tag{1}$$

which represents how much the element x_i is responsible for the prediction $f(\mathcal{X})$. In practice, one can compute the L2-norm of g_i and normalize over all the L2-norms of the input sequence to obtain a score $p_i \in [0, 1]$, which represents the contribution (attribution) of x_i to a positive prediction for $\mathcal{X}$.

Based on the technique, we devise the following training task for keyword prediction. Denote by N the number of paraphrase pairs in the training set, and (s_i, t_i) the source sentence and the target sentence of the i^{th} pair, respectively, $1 \leq i \leq N$. We first construct N positive data points (i.e., each data point corresponds to a paraphrase pair) where the i^{th} data point consists of s_i, the special token `[SEP]` and t_i, sequentially, i.e., $(s_i, \texttt{[SEP]}, t_i)$. Because during inference the target sentence is unknown, we construct another N positive data points $(s_i, \texttt{[SEP]}, s_i)$ for training. Then for each s_i, we randomly select two different target sentences t_{i_1} and t_{i_2}, such that $i_1 \neq i$ and $i_2 \neq i$, and form two negative data points $(s_i, \texttt{[SEP]}, t_{i_1})$ and $(s_i, \texttt{[SEP]}, t_{i_2})$. As such there are in total $2N$ positive and $2N$ negative data points. Then we fine-tune a BERT model using the $4N$ data points to predict whether each data point is a paraphrase pair. After fine-tuning, given a new data point consisting of a source sentence and its paraphrase (the source sentence itself during inference), we first compute the output in the forward pass, and then compute the attribution scores of all the input words in the backward pass. Since the attribution score reflect how much each word contributes to the paraphrase prediction, the words with higher scores are more likely to be the keywords that capture the common semantics of the two sentences. For keyword prediction, we just use the attribution scores of the words in the source sentence. Figure 1 shows the inference process for predicting the keywords of the source sentence in the running example. We observe the five words "good", "workouts", "lose", "belly" and "fat" are more likely to be the keywords.

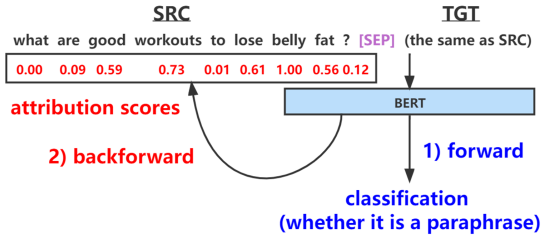

Figure 1: Stage One of PGKPR.

4 Stage Two: Multi-task Learning for Paraphrase Generation

Figure 2 shows the overview of the second stage. The model is trained simultaneously with three tasks: reconstruction of keywords and POS tags, contrastive learning for distinguishing paraphrase pairs from others, and paraphrase generation.

4.1 Task 1: Reconstruction of Keywords and POS Tags

Given a source sentence s_i, the task is learning to reconstruct the keywords of s_i and the POS tags of all the words, so that both the key semantics of s_i and the relations between the keywords are captured in the latent feature. After obtaining the attribution scores of s_i in the first stage, we consider each score as the probability that the corresponding word is a keyword. Then we flip a coin for each word with the probability and identify the final keywords of s_i. In this way, we flexibly set the keywords in each sentence and avoid overfitting the training set to some extent. On the left part of Figure 2, we observe that the five words in red are computed as the keywords based on the probabilities. Then we form an input token sequence TS_{s_i} as a two-part representation based on the perturbation to s_i, which contains the POS-tag information of s_i while also distinguishing the keywords from the non-keywords, as follows. The first part of TS_{s_i} is a perturbation of s_i, where the keywords are preserved in the sequence and the non-keywords are replaced by their corresponding POS tags. The second part is another perturbation of s_i in the other way round, where the non-keywords are preserved and the keywords are replaced by their corresponding POS tags. There is a special token `[SEP]` connecting the two parts. The idea is to use the first part to emphasize the keywords and their relations

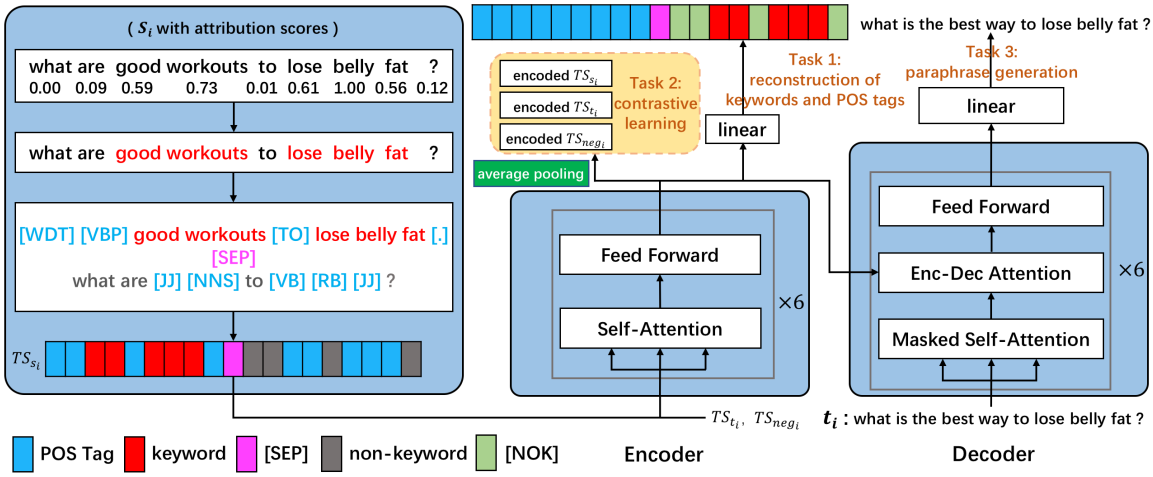

Figure 2: Stage Two of PGKPR: the Transformer-based model with three learning tasks for paraphrase generation.

(via POS tags), and use the second part to emphasize the POS information of the keywords and the content information of non-keywords. The process to form TS_{s_i} for the running example is depicted in the left part of Figure 2.

Then we feed TS_{s_i} into the Transformer's encoder. Essentially, we want to produce the encodings that preserve the POS and semantic feature of keywords, and only preserve the POS feature of non-keywords[1] of s_i. We devise the following task to achieve the goal, which attempts to reconstruct the keywords and POS tags of s_i. For each encoding in the first part of TS_{s_i}, we train it to predict the POS tag of the corresponding word in s_i. As such the output encodings could learn the syntactic feature of s_i and particularly the relations between the keywords. The encoding of the special token `[SEP]` learns to reconstruct itself, so that it still separates the output encodings into two groups with different emphasis. For each encoding in the second part of TS_{s_i}, if it corresponds to the POS tag of a keyword, we use it to reconstruct the keyword so that the encoding learns the semantic of the keyword; otherwise, it corresponds to a non-keyword and we use it to predict a special token `[NOK]` (representing "non-keyword"), which forces the encoding to downplay the semantic feature of the non-keyword and learn more the position feature. The task is depicted in the middle part of Figure 2. Denote by y_j^i the target token of the j^{th} token of TS_{s_i} and by $p(y_j^i)$ the predicted probability, the reconstruction loss function $\mathcal{L}_{rec}^i$ for s_i is computed using cross-entropy:

$$\mathcal{L}_{rec}^i = -\frac{1}{2l_s+1}\sum_{j=1}^{2l_s+1} p(y_j^i)\log(p(y_j^i)), \quad (2)$$

where l_s and $2l_s+1$ are the length of s_i and TS_{s_i}.

4.2 Task 2: Contrastive Learning for Distinguishing Paraphrase Pairs from Others

Inspired by (Yang et al., 2021; Pan et al., 2021), we devise a contrastive learning task to distinguish the syntactic and semantic features of paraphrase pairs from non-paraphrase pairs, so that the learned encodings of a source sentence are more discriminative. The general principle of contrastive learning (Chen et al., 2020) is to minimize the distances between the data point and the positive counterparts while maximizing the distances between the data point with the negative counterparts, in the latent space.

In our task, for each s_i, we use the corresponding target sentence t_i as the positive counterpart and use all other sentences in the same batch as the negative counterparts. We denote the negative counterparts by $\{neg_i \in B | neg_i \notin \{s_i, t_i\}\}$, where B is a minibatch containing (s_i, t_i). For each counterpart, we form an input token sequence by concatenating the original sentence, `[SEP]` and the POS tag sequence of the sentence. By doing this, we can not only make the input length of the counterparts conform with TS_{s_i}, but also keep both the syntactic and semantic information of the counterpart sentences. As such, the encoding of s_i could learn more discriminative features pertaining to the keywords and their relations. The token sequences

[1]The semantic feature of non-keywords is captured in the generation task.

of t_i and neg_i are denoted by TS_{t_i} and TS_{neg_i}, respectively, as depicted in the middle part of Figure 2. We apply average pooling over the token encodings and obtain the encoded TS_{s_i}, TS_{t_i} and TS_{neg_i}. Note that we don't perform perturbation to the counterpart sentences because the average pooling layer would eliminate the effect of perturbation. Denote by e_{s_i}, e_{t_i} and e_{neg_i} the corresponding encodings, the contrastive loss function $\mathcal{L}^i_{con}$ for s_i is computed as follows,

$$\mathcal{L}^i_{con} = -\log \frac{exp(\frac{e_{s_i} \cdot e_{t_i}}{\tau})}{exp(\frac{e_{s_i} \cdot e_{t_i}}{\tau}) + \sum\limits_{neg_i \in B}^{neg_i \notin \{s_i, t_i\}} exp(\frac{e_{s_i} \cdot e_{neg_i}}{\tau})}, \tag{3}$$

where $\cdot$ denotes the dot product and τ is the temperature parameter.

4.3 Task 3: Paraphrase Generation

The last learning task is to generate the paraphrase sentence on the decoder side, which is depicted on the right part of Figure 2. All the token encodings output by the encoder participate into the computation of the encoder-decoder attention layer, so that the decoder can retrieve the information pertaining to both the key semantics of the source sentence via the encodings of the keywords and the relations between the keywords via the encodings of the POS tags. Denote by t^i_j the j^{th} word in the target sentence t_i, the generation loss function $\mathcal{L}^i_{gen}$ for s_i is computed using the sum of negative log-likelihood as follows,

$$\mathcal{L}^i_{gen} = -\sum_{j=1}^{l_t} \log p(t^i_j \mid s_i, \{t^i_0, t^i_1, \ldots, t^i_{j-1}\}), \tag{4}$$

where l_t is the length of the target sentence.

4.4 The Objective Function

The final objective function of PGKPR is the linear combination of the loss functions in the three learning tasks, which is computed as follows,

$$\mathcal{L}^i = \lambda_1 \mathcal{L}^i_{rec} + \lambda_2 \mathcal{L}^i_{con} + \mathcal{L}^i_{gen}, \tag{5}$$

where λ_1 and λ_2 are the two hyperparameters.

5 Performance Evaluation

We implement all the models using Pytorch 1.4 and run all experiments on a Centos machine installed with Tesla V100.

5.1 The DataSets and Evaluation Metrics

We conduct the experiments on two benchmark datasets for paraphrase generation, which are Quora[2] and MSCOCO (Lin et al., 2014). The Quora dataset contains duplicated questions raised by real users, in which each data point consists of a source question and a target question with the similar meaning. The MSCOCO dataset contains images and the corresponding captions annotated by humans. Since each image has five captions, we randomly choose one of them as the source sentence and use the other four as the targets. As such each image brings four pairs of paraphrases.

Following (Gupta et al., 2018; Fu et al., 2019), we split the pre-processed datasets into the training and testing set. For the Quora dataset, there are 100K training paraphrase pairs and 20K testing pairs. The sentences are truncated or zero-padded to the same length 17 to facilitate batch training. For the MSCOCO dataset, there are 93K training pairs and 20K testing pairs. The sentence length is set to 16.

For the main results, we use the commonly-adopted metrics BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) to evaluate the models, as they are proved to correlate with human judgement well (Li et al., 2018; Fu et al., 2019). We report the metrics of 1-4 grams in BLEU, 1-2 grams in ROUGE and ROUGE-L.

5.2 The Comparative Models

Although paragraph generation draws lots of attention, few studies have tried to explicitly preserve the keywords as well as their relations in the source sentence. Among the existing studies, we identified two models that are closely related to ours.

The first model is IANet (Su et al., 2021), which proposes the Primary/Secondary Identification (PSI) algorithm to separate the primary and secondary content of a source sentence. We implemented the two variants mentioned in the paper[3], IANet+X and IANet+S, which use the rule-based method and the pre-training method to identify the primary content. Both variants rely on a manually-determined threshold to separate the primary and secondary content.

The second model is LBOW (Fu et al., 2019), which samples a latent bag of words from the en-

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

[3] The authors have not released the source code.

Quora							
Models	**B-1**	**B-2**	**B-3**	**B-4**	**R-1**	**R-2**	**R-L**
Residual-LSTM (Prakash et al., 2016)	55.06	40.73	31.41	25.06	56.92	32.70	54.37
Transformer (Vaswani et al., 2017)	57.26	43.44	34.20	27.79	58.89	34.92	56.16
LBOW-Topk (Fu et al., 2019)	55.94	42.02	32.64	26.10	58.60	34.33	56.17
LBOW-gumbel (Fu et al., 2019)	55.82	41.82	32.48	25.96	58.09	33.88	55.59
IANet+X (Su et al., 2021)	57.69	43.78	34.30	27.70	59.00	35.15	56.43
IANet+S (Su et al., 2021)	57.72	43.74	34.24	27.65	59.03	35.10	56.41
PGKPR	**58.89**	**45.08**	**35.69**	**29.23**	**60.94**	**36.69**	**58.16**
PGKPR-ref	58.89	45.07	35.68	29.24	60.82	36.58	58.02
PGKPR-PSI+X	58.37	44.21	34.78	28.31	58.32	35.18	56.36
PGKPR-PSI+S	58.46	44.22	34.77	28.27	59.44	35.09	56.39

MSCOCO							
Models	**B-1**	**B-2**	**B-3**	**B-4**	**R-1**	**R-2**	**R-L**
Residual-LSTM (Prakash et al., 2016)	71.67	49.88	34.57	24.50	41.85	15.74	37.76
Transformer (Vaswani et al., 2017)	71.41	50.86	35.42	25.14	41.60	15.52	37.46
LBOW-Topk (Fu et al., 2019)	72.62	51.00	35.53	25.30	42.16	16.09	38.20
LBOW-gumbel (Fu et al., 2019)	72.41	51.85	35.51	25.16	42.20	16.05	38.15
IANet+X (Su et al., 2021)	70.43	49.50	34.09	23.95	40.76	14.80	36.78
IANet+S (Su et al., 2021)	71.46	50.93	35.29	24.80	41.37	15.36	37.40
PGKPR	**72.67**	**52.55**	**37.22**	**26.70**	**42.49**	**16.31**	**38.25**
PGKPR-ref	72.67	52.66	37.34	26.87	42.46	16.16	38.16
PGKPR-PSI+X	70.61	49.99	34.68	24.46	41.39	15.15	37.22
PGKPR-PSI+S	72.03	51.73	36.37	25.95	42.18	15.89	37.82

Table 2: The main results on Quora and MSCOCO. All the numbers are obtained from either implementing the corresponding models, if the source code is not available, or from running the source code released by the authors.

coder to assist the paraphrase generation. The words in the latent bag could be considered to have similar semantics with the keywords in the source sentence, and therefore the model is related to ours. We obtained the code released by the authors[4] and evaluate the two variants LBOW-Topk and LBOW-Gumbel. The former directly chooses the most k probable words from the encoder, and the latter samples randomly from the BOW distribution with gumbel reparameterization.

We include another two models as baselines. The first model is Residual-LSTM (Prakash et al., 2016), which is the very first study that applies deep learning to paraphrase generation. The second model is the original Transformer (Vaswani et al., 2017). We train it directly with the paraphrase pairs in the simple sequence-to-sequence manner.

5.3 The Hyperparameters

Both the encoder and decoder of PGKPR have 6 layers and each layer uses 8 attention heads. The embedding size is set to 512. When training, we set dropout rate to 0.1, learning rate to 0.0001, and use Adam for optimization. The batch size is set to 128. After tuning, we set λ_1 and λ_2 in the objective function to 1 and 0.1 for the Quora dataset, and set to 1 and 1 for the MSCOCO dataset, respectively.

[4] https://github.com/FranxYao/dgm_latent_bow

5.4 Main Results

The main results are reported in Table 2. We observe that our PGKPR model outperforms all the comparative models by a notable margin on both datasets.

To further justify the effectiveness of PGKPR, we implemented two additional variants of the model. The first variant is PGKPR-ref, which uses the true paraphrase pairs to identify the keywords in the first stage during inference. Remember that in PGKPR we concatenate a source sentence with itself during inference, since the target sentence is unknown. However, the upper bound of the performance should be achieved when the target sentence is disclosed, i.e., using the true pair of a source sentence and a target sentence to predict the keywords. In Table 2 we observe that PGKPR-ref does not always outperform PGKPR and the overall performance of PGKPR is very close to PGKPR-ref. The reason is that we add the pairs of two source sentences in the training set (see the second paragraph of Section 3), so that PGKPR generalizes well at inference time when the target sentence is unknown.

The second variant is PGKPR-PSI, which uses the PSI algorithm in IANet to identify the keywords. Following (Su et al., 2021), we imple-

Quora							
Models	**B-1**	**B-2**	**B-3**	**B-4**	**R-1**	**R-2**	**R-L**
PGKPR	**58.89**	**45.08**	**35.69**	**29.23**	**60.94**	**36.69**	**58.16**
PGKPR w/o $\mathcal{L}_{con}$	58.63	44.74	35.26	28.61	60.31	36.45	57.73
PGKPR w/o $\mathcal{L}_{rec}$	58.33	44.27	34.87	28.42	59.91	35.35	57.04
PGKPR w/o $\mathcal{L}_{con}$ and $\mathcal{L}_{rec}$	58.1	43.89	34.42	27.91	58.90	35.35	56.84

MSCOCO							
Models	**B-1**	**B-2**	**B-3**	**B-4**	**R-1**	**R-2**	**R-L**
PGKPR	**72.67**	**52.55**	**37.22**	**26.70**	**42.49**	**16.31**	**38.25**
PGKPR w/o $\mathcal{L}_{con}$	72.29	51.99	36.71	26.29	42.34	16.12	38.04
PGKPR w/o $\mathcal{L}_{rec}$	72.12	51.94	36.60	26.18	42.33	16.00	37.99
PGKPR w/o $\mathcal{L}_{con}$ and $\mathcal{L}_{rec}$	71.87	51.59	36.28	25.85	42.26	15.95	37.97

Table 3: Ablation Study.

mented PGKPR-PSI+X and PGKPR-PSI+S, which are the counterparts of IANet+X and IANet+S. In Table 2 we observe two points. First, the PGKPR-PSI variants perform worse than the original PGKPR. Since the only difference between them is the mechanism for keyword identification, we may conclude that our model-based identification strategy is more suitable for extracting keywords from a source sentence. Second, the PGKPR-PSI variants perform better than the IANet counterparts on almost all metrics. Since both models use PSI to identify the keywords, the results show the effectiveness of multi-task learning in the PGKPR model.

5.5 Ablation Study

We conduct an ablation study to show the effect of the reconstruction loss and the contrastive loss in the multi-task learning. In particular, we remove from the original PGKPR model only the contrastive loss, only the reconstruction loss and both losses, respectively, which results in three ablation models PGKPR w/o $\mathcal{L}_{con}$, PGKPR w/o $\mathcal{L}_{rec}$ and PGKPR w/o $\mathcal{L}_{con}$ and $\mathcal{L}_{rec}$. The results are reported in Table 3. We observe a significant performance drop after removing the losses. Specifically, removing the reconstruction loss results in a larger performance drop than removing the contrastive loss. This justifies the motivation of the current study, i.e., capturing the key semantics and the relations between the keywords in the source sentence should benefit paraphrase generation.

5.6 Comparing with the PSI Algorithm

Only our PGKPR model and the IANet model (Su et al., 2021) explicitly identify the keywords from a source sentence. While IANet uses a rule-based algorithm PSI to identify the keywords, PGKPR adopts the purely data-driven approach based on

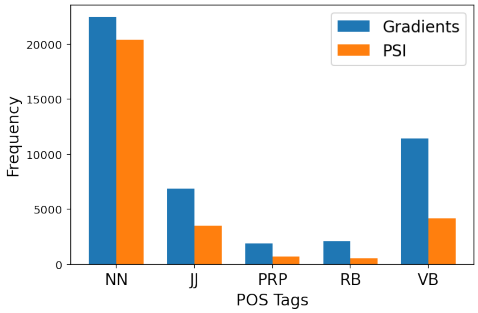

Figure 3: The frequency distribution of the POS tags on Quora.

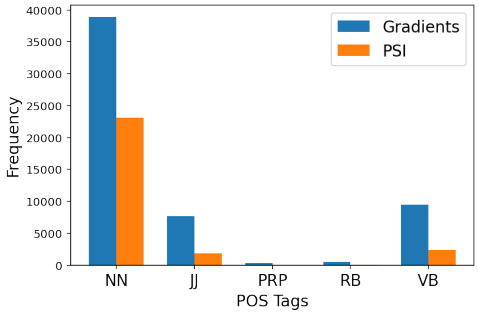

Figure 4: The frequency distribution of the POS tags on MSCOCO.

a prediction attribution technique that computes the gradients. It is thus interesting to compare the keywords identified by the two methods.

To this end, we first extract the keywords from the dataset using the two methods, respectively, and then plot the frequency distribution of the POS tags pertaining to the keywords. For PSI, we set the threshold of the PSI score to separate the primary and secondary content when the IANet-S model achieves the best performance on the testing set. For PGKPR, the keywords are selected with the probabilities calculated from the L2-norms of their gradients (see the first paragraph of Section 4.1).

	Quora	MSCOCO
Source	what are good workouts to lose belly fat ?	a woman with a toothbrush in her mouth
Target	what is the best way to lose belly fat ?	a person standing with a toothbrush in their mouth
Residual-LSTM	what are the best ways to lose belly fat ?	a woman with a toothbrush in her mouth
Transformer	what are some good ways to get rid of belly fat ?	a bunch of food on a table outside
LBOW-Topk	how can i reduce my belly fat ?	a woman is holding a toothbrush in her mouth
IANet+S	what are some workouts to lose weight ?	a woman with a toothbrush in her mouth
PGKPR	what are some good exercises to get rid of belly fat ?	a woman brushing her teeth with a tooth brush

Table 4: Case Study.

The results are shown in Figure 3 and 4, which are the plots on Quora and MSCOCO, respectively. On the X-axis, we use five POS tags, namely, NN, JJ, PRP, RB and VB, which correspond to nouns, adjectives or numerals, pronouns, adverbs and verbs, respectively. It is of the common sense that the words of these POS tags preserve the key semantics of a sentence, and thus we refer to them as the key POS tags. The Y-axis shows the number of each POS tag extracted by the two methods. We observe that on both datasets our gradient-based method extracts more key POS tags than the PSI algorithm does. The results may explain why the original PGKPR model performs better than the PGKPR-PSI variants in Table 2.

5.7 Case Study

In Table 4, we show the generated paraphrases of the five models for two source sentences in the Quora and MSCOCO dataset, respectively. On the left part, we see PGKPR captures the keywords "good workouts" and "lose belly fat", and uses the synonyms "exercises" and "get rid of" in the paraphrase. Other models are generally good, but the paraphrases are not as accuracy and diverse as ours. The sentence produced by IANet+S fails to capture the keyword "belly". On the right part, PGKPR not only captures the key semantics of the source sentence, but also changes the syntax structure. All other models either fail to capture the key semantics or produce a paraphrase syntactically similar to the source sentence. The sentence produced by IANet+S simply repeats the source sentence.

5.8 User Study

We conduct a user study on the quality of the paraphrases generated by the compared models. For LBOW and IANet, we choose the variants with overall better performance in Table 2, namely, LBOW-Topk and IANet+S. As such there are five models for this study. The evaluated metrics are *Fluency*, *Accuracy*, and *Diversity*. Fluency measures whether a sentence is grammatically correct. Accuracy measures whether the semantics of a generated sentence comply with that of the corresponding source sentence. Diversity measures whether a generated sentence differs from the corresponding source sentence in terms of syntax structure.

Models	**Fluency**	**Accuracy**	**Diversity**
Residual-LSTM	1.49	1.14	0.8
Transformer	1.7	1.33	1.11
LBOW-Topk	1.55	1.21	0.85
IANet+S	1.68	1.37	0.97
PGKPR	**1.79**	**1.5**	**1.29**
Target	1.85	1.59	1.47

Table 5: The results of human evaluation. Statistical significance between PGKPR and others is computed with a 2-tailed Student's t-test; $p\text{-}value < 0.05$.

We invite ten Master's students to rate the generated paraphrases. We randomly choose 100 source sentences from the testing sets (50 for Quora and 50 for MSCOCO) and generate the paraphrases for each sentence using the five models. We replicate three times each source sentence and its five paraphrases and obtain 1,500 pairs of paraphrases. We randomly assign the paraphrases to the 10 students, so that each student is assigned with 150 different pairs. We ask the students to rate each generated paraphrase on the three metrics on a scale between 0 to 2, where a higher score means better quality. Then we compute the average scores for each model and the statistical significance between PGKPR and other models. The results are reported in Table 5, where "Target" means the target sentence. We observe that PGKPR performs the best on the three metrics among the models and the difference between PGKPR and each model is statistically significant ($p\text{-}value < 0.05$), verified

using a 2-tailed Student's t-test. The results justify the effect of learning simultaneously the keywords and the relations between them and the design of the multiple learning tasks in PGKPR.

6 Conclusion

We propose a new model with multi-task learning for paraphrase generation. The motivation is to simultaneously capture the key semantics of a source sentence and the relations between the keywords. The proposed model, PGKPR, has two stages. In the first stage, PGKPR leverages a data-driven technique to identify the possible keywords in the source sentence. In the second stage, PGKPR adopts the Transformer model and devises three learning tasks, including 1) reconstructing the keywords and the POS tags of all words in the source sentence, 2) contrastive learning for distinguishing the latent features of the paraphrase pair from others, and 3) generating the paraphrase sentence. We conduct extensive experiments to show the superior performance of PGKPR over comparative models, as well as the effect of the keyword identification strategy and the multiple learning tasks. Our future research would focus on how to apply the model in the current study to controllable paraphrase generation and produce more diverse sentences.

Acknowledgement

This work is supported by the grant from the National Natural Science Foundation of China (Grant No. 61977026).

References

Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017. Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 159–168.

S Bach, A Binder, G Montavon, F Klauschen, KR Müller, and W Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7):e0130140.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. 2010. How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11:1803–1831.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155.

Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. 2016. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks*, pages 63–71. Springer.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase generation with latent bag of words. *Advances in Neural Information Processing Systems*, 32:13645–13656.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033.

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, Volume 1 (Long Papers)*, pages 1875–1885.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016a. Visualizing and understanding neural models in nlp. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016b. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Kathleen McKeown. 1979. Paraphrasing using given and new information in a question-answer system. In *17th Annual Meeting of the Association for Computational Linguistics*, pages 67–72.

Marie Meteer and Varda Shaked. 1988. Strategies for effective paraphrasing. In *Coling Budapest 1988 Volume 2: International Conference on Computational Linguistics*.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 244–258.

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*, pages 311–318.

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

Chris Quirk, Chris Brockett, and William B Dolan. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 142–149.

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6033–6039.

Yixuan Su, David Vandyke, Simon Baker, Yan Wang, and Nigel Collier. 2021. Keep the primary, rewrite the secondary: A two-stage approach for paraphrase generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 560–569.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2010. Paraphrase generation as monolingual translation: Data and evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*.

Haoran Yang, Wai Lam, and Piji Li. 2021. Contrastive representation learning for exemplar-guided paraphrase generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4754–4761.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.