Extract, Transform and Fill: A Pipeline Model for Question Paraphrasing Based on Templates

Yufan Gu[12], **Yuqiao Yang**[13], **Zhongyu Wei**[1*]

[1]School of Data Science, Fudan University
[2]Alibaba Group, China
[3]Department of Information and Communication, School of Engineering, Tokyo institute of Technology
aleck16@163.com, yyqfaust@gmail.com, zywei@fudan.edu.cn

Abstract

Question paraphrasing aims to restate a given question with different expressions but keep the original meaning. Recent approaches are mostly based on neural networks following a sequence-to-sequence fashion, however, these models tend to generate unpredictable results. To overcome this drawback, we propose a pipeline model based on templates. It follows three steps, a) identifies template from the input question, b) retrieves candidate templates, c) fills candidate templates with original topic words. Experiment results on two self-constructed datasets show that our model outperforms the sequence-to-sequence model in a large margin and the advantage is more promising when the size of training sample is small.

1 Introduction

Paraphrase means sentences or phrases that convey the same meaning with different expressions. Popular tasks about paraphrases are paraphrase identification (Yin and Schütze, 2015), paraphrase generation (Li et al., 2018; Gupta et al., 2018), sentence rewriting (Barzilay and Lee, 2003), etc. As a special case of paraphrase generation, question paraphrasing (McKeown, 1983) aims to restate an input question. It can be applied in a question answering system for the expansion of question set to enhance the coverage of candidate answers. Besides, it is able to probe the need of users within an interactive system by rephrasing questions.

Traditional approaches for paraphrase generation are mostly based on external knowledge, including manually constructed templates (McKeown, 1983), or external thesaurus (Hassan et al., 2007). The generated paraphrases are usually fluent and informative. However, it is very time-consuming to construct templates by human and external thesaurus are always absent for some languages. Recently, researchers start to use neural network based approaches by formulating the generation task in a fashion of sequence-to-sequence (Sutskever et al., 2014; Bahdanau et al., 2014; Prakash et al., 2016). However, these models tend to "lose control" generating some unpredictable results.

Original Question
请帮我查一下卡片的开户行
Please help me check the card's bank.
Paraphrase Questions
我想知道卡片的开户行
I would like to know the card's bank
您好,请帮我查询一下卡片的开户行
Hi, please help me checkthe card's bank
卡片的开户行请帮我查询一下
The card's bank, please help me check it
卡片的开户行能帮我查询一下吗?
The card's bank, can you help me check it?

Table 1: Example of an question and its paraphrases. Underlined phrases are topic words and others are templates.

In order to alleviate the uncertainty in sequence-to-sequence model, Cao et al. (2018) propose to search for similar sentences as soft template to back up the neural generation model in the scenario of text summarization. With this inspiration, we also try to bridge neural-based models and template-based approaches for question paraphrasing. An example of question paraphrasing can be seen in Table 1. We have two observations. First, words in a question can be easily divided into two types, namely, topic words and template words. Template words define the information need of the question while topic words are related to some specific entities or events. Second, for a pair of paraphrase questions, they tend to share the

* Corresponding author

Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 109–114

same topic words while template words are different. Motivated by these two observations, we try to identify template and topic words in the original question and construct paraphrase questions by considering these two parts separately.

In this paper, we propose a template-based framework to generate question paraphrase in a pipeline. The framework mainly includes three components, namely template extraction, template transforming and template filling. The contribution of our paper is three-fold.

- First, we propose a pipeline model to identify template and topic words from a question and generate the question paraphrases via template transforming and filling.
- Second, we construct two datasets for question paraphrasing collected from two domains, namely financial domain and automotive domain. All topic words are labeled in questions. The dataset is available here [1]
- Third, extensive experiments are performed on the self-constructed dataset to evaluate the effectiveness of our pipeline model. Results show that our model outperforms the state-of-the-art approach in a large margin.

2 Datasets Description

Two datasets are collected and annotated for question paraphrasing, including banking service questions from the financial domain and sales service questions from the automotive domain. The annotation consists of two parts. First, we classify questions into different clusters so that questions in each cluster share the same meaning. Second, we label template and topic words in each question. The number of question clusters for the financial domain and automotive domain are 2,589 and 526 respectively. Note that, for each cluster in financial dataset, we have 5 paraphrasing questions and for each cluster in automotive dataset, we have 4 paraphrasing questions.

The annotation of question cluster is performed by experts in the two domains, while two student annotators are hired for the labeling of the templates. For the template identification, annotators are instructed that the template part should be generalized, which means that the question will be readable if we replace the topic words with other similar content.

The agreement between annotators for template identification is 0.558 and 0.568 for the domain of finance and automotive respectively. Further observations on the annotation results of template identification show that even if templates identified by the two annotators are different, both templates can be reasonable. We therefore construct two versions of datasets for experiments. One keeps both annotations (union) and the other includes questions with same labels from annotators (intersection). The statistics of our datasets can be seen in Table 2.

Statistics	financial		automotive	
	inter.	union	inter.	union
# of questions	7,218	12,938	1,195	2,103
# of templates	6,574	17,300	1,184	2,998
# of vocab.	908	1,100	656	907
# of template vocab.	325	528	144	303
# of topic vocab.	869	1,063	620	873

Table 2: Statistics of annotated datasets for question paraphrasing. *inter.* is short for intersection; *vocab.* is short for vocabulary; vocabulary here means unique tokens.

3 Proposed Model

Given the input question q, question paraphrasing system aims to generate questions with the same meaning but different expressions. Our proposed template-based model follows a pipeline fashion. It includes three main components, namely, template extraction, template transforming and template filling. The template extraction module classifies words in the input question into template part and topic part. Template transforming module searches for candidate templates for paraphrasing. Finally template filling module fills in the slots of the retrieved templates with topic words. And we take two training approaches, one is separate training and the other is joint training. A running example can be seen in Figure 1.

3.1 Template Extraction

Take a question as input, template extraction module classifies words into template and topic ones. We treat the problem as a supervised sequence labeling task and modify the classical *BIO* tagging strategy to fit our scenario. Specifically, we use "O" to specify the template part, and treat "B" and "I" as the topic part. As Bi-LSTM has been

[1] http://www.sdspeople.fudan.edu.cn/zywei/data/paraphrase.zip

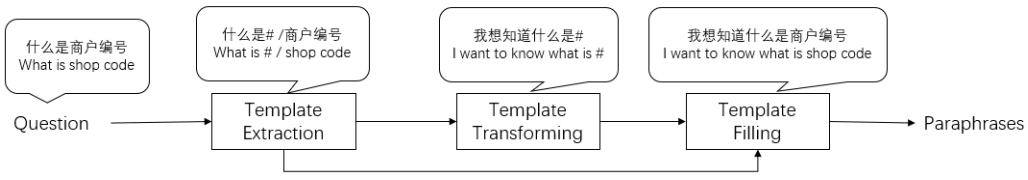

Figure 1: The overview of the proposed framework.

proved to be effective for the task of sequence labeling (Ma and Hovy, 2016), we also utilize such structure for template extraction. Cross-entropy (CE) is used for training and the loss is J_{TE}.

3.2 Template Transforming

Take the extracted template from previous module as input, template transforming module searches for candidate templates for paraphrasing. We utilize a retrieval-based approach to search for candidate templates. We first build an index for all the templates in our dataset. Then we use a score function (e.g. cosine similarity) to evaluate the similarity between original template and candidate templates to find out the most similar template.

To better represent our template, we train a sequence-to-sequence model with attention for template transforming. For each template, the hidden state resulted from the encoder is used as its representation. Note that, we also tried the generation results directly, however, preliminary experiment results showed the model performs poor. The loss for training seq-to-seq model is J_{TT}.

3.3 Template Filling

Take a candidate template and topic words as input, template filling module fills each slot in the template with topic words to form a new question. In practice, we use two encoders to encode subsequence of topic part and candidate template separately. Then we concatenate topic representation and candidate representation, and put them into a classifier to predict the position of the slot for the particular topic word. Cross-entropy is used here for training and loss is denoted by J_{TF}.

3.4 Training

We study two different approaches for the training of our pipeline model, namely *separate training* and *joint training*. For separate training, we train three modules (template extraction, template transforming and template filling) separately and combine them together for the whole framework. We can also train them together to ease the error propagation problem resulted from separate training. The loss function here is the sum of each module.

$$J(\theta) = J_{TE}(\theta) + J_{TT}(\theta) + J_{TF}(\theta) \quad (1)$$

4 Experiments

4.1 Experimental Setup

We test our model on datasets described in Section 2. Both datasets are divided into training, validation and test with split ratio of 7:2:1. We use Adam as our optimization method and set the learning rate as 0.0001. We set the dimension of hidden state as 128. For padding, we set the max length as 64. We use BERT-Chinese tokenizer(Devlin et al., 2018) to separate characters.

For the general evaluation, we evaluate the quality of the generated paraphrase questions. BLEU-1, BLEU-2, BLEU-3, BLEU-4((Papineni et al., 2002)) are used as evaluation measures. Three models are compared.

seq2seq (Bahdanau et al., 2014) uses an encoder-decoder structure with attention for generation.

ours (separate) this is our pipeline model consisting of three modules. Each module is trained separately.

ours (joint) this is our pipeline model consisting of three modules and joint training is used.

4.2 Overall Evaluation

The overall experiment results can be seen in Table 3. Both of our pipeline models based on template outperform sequence-to-sequence model in a large margin on all the four datasets in terms of all the four metrics. The performance of *ours (joint)* is better than that of *ours (separate)* which indicates that joint training is effective for the pipeline model. The performances of all three models on the *union* set are better than their counter-part on

Dataset	Model	BLEU-1		BLEU-2		BLEU-3		BLEU-4	
		intersection	union	intersection	union	intersection	union	intersection	union
Financial	seq2seq	0.658	0.803	0.577	0.741	0.504	0.683	0.444	0.630
	ours (separate)	0.863	0.892	0.808	0.832	0.753	0.772	0.698	0.716
	ours (joint)	**0.873**	**0.902**	**0.827**	**0.857**	**0.782**	**0.812**	**0.739**	**0.770**
Automotive	seq2seq	0.581	0.771	0.526	0.723	0.482	0.684	0.441	0.648
	ours (separate)	0.826	**0.850**	0.757	0.777	0.701	0.713	0.650	0.654
	ours (joint)	**0.859**	0.849	**0.808**	**0.790**	**0.763**	**0.738**	**0.720**	**0.690**

Table 3: The overall performance of different models on four datasets from two domains (**bold** number in each column is the best performance on that dataset).

the *intersection* set. This is probably because the size of training samples are larger in the *union* set. Moreover, the sequence-to-sequence model is more sensitive to the size of training set, while our template-based model can achieve comparable performance on both sets.

4.3 Further Analysis for Transfer Learning

In addition to the overall performance of our pipeline model, we also analyze its performance for transfer learning. Since we have datasets from two domains, and the financial one is much bigger than the one from automotive domain. It is natural to train the model in the bigger dataset and transfer it to the domain with less training data. We thus report the experiment results for transfer learning from financial domain to the automotive one. Here, we compare three settings for the training of our model.

f2a: Model is trained on the financial dataset.

a2a: Model is trained on the automotive dataset only. It is the same joint model as we used in the previous section.

f+a2a: Model is pre-trained on the financial dataset and then fine-tuned on the automotive dataset.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Seq2Seq(f2a)	0.251	0.167	0.110	0.085
Seq2Seq(a2a)	0.581	0.526	0.482	0.441
Seq2Seq(f+a2a)	0.715	0.661	0.619	0.580
ours (f2a)	0.796	0.722	0.656	0.598
ours (a2a)	0.859	0.808	0.763	0.720
ours (f+a2a)	**0.881**	**0.835**	**0.791**	**0.747**

Table 4: Transfer learning performance of our pipeline model on the intersection datasets (**bold** number in each column is the best performance on that dataset).

Performance for transfer learning can be seen in Table 4. The performance of *ours (f2a)* that directly applies the model trained on financial domain to automotive domain is better than the performance of *Seq2Seq*. This indicates that template-based model is easier to be transferred from one domain to the other. *ours (f2a)* is worse than *our (a2a)*, this is reasonable because there is a gap between dataset, such as different vocabularies and different templates. The performance of *ours (f+a2a)* is better than *ours (a2a)*. This shows that fine-tuning on the target domain can further improve the model. The results on *Seq2Seq (f2a)*, *Seq2Seq (a2a)* and *Seq2Seq (f+a2a)* show the same trend. The experiment we have done in this part also gives us a new way to improve the performance of our model when the size of target dataset is limited.

5 Related Work

There are two lines of research for paraphrase generation including knowledge based ones and neural network based ones. Some researchers provide rules (Bhagat and Hovy, 2013) or corpus including knowledge (Fader et al., 2013; Ganitkevitch et al., 2013; Pavlick et al., 2015). Other researchers try to make use of templates (Berant and Liang, 2014), semantic information (Kozlowski et al., 2003) and thesaurus (Hassan et al., 2007) for paraphrase generation.

Rush (2015) have applied Seq2Seq model with attention mechanism for text summarization. Prakash (2016) employ a residual net in Seq2Seq model to generate paraphrases. Cao (2017) combine a copying decoder and a generative decoder for paraphrase generation. Cao(2018) try to utilize template information to help text summarization, however, the template is vague in that paper. We hope to utilize the special structure of question and extract the template explicitly from questions.

6 Conclusion

In this paper, we proposed a template-based framework for paraphrase question generation including three components, template extraction, template transforming and template filling. We identify template and topic words via template

extraction and generate paraphrase questions via template transforming and filling. Experiment results on two self-constructed datasets from two domains showed that our pipeline model outperforms seq2seq model in a large margin.

Acknowledgments

Thanks for the constructive comments from anonymous reviewers. This work is partially funded by National Natural Science Foundation of China (No. 61751201), National Natural Science Foundation of China (No. 61702106) and Shanghai Science and Technology Commission (No. 17JC1420200, No. 17YF1427600 and No. 16JC1420401).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. 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 (Volume 1: Long Papers)*, volume 1, pages 1415–1425.

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 152–161.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1608–1618.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. 2007. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 410–413.

Raymond Kozlowski, Kathleen F McCoy, and K Vijay-Shanker. 2003. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 1–8. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

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

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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 425–430.

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. *arXiv preprint arXiv:1610.03098*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Wenpeng Yin and Hinrich Schütze. 2015. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911.