Paraphrasing via Ranking Many Candidates

Joosung Lee
Kakao Enterprise, South Korea

Kakao**enterprise**

Motivation

- Paraphrasing sentences can be utilized in various NLP applications.
- It is difficult to ensure that one generation method always generates the best paraphrase in various domains.
- Even a good system does not paraphrase all sentences with good quality.
- We focus on finding the best candidate from multiple candidates, rather than assuming that there is only one combination of generative models and decoding options.

Problem Setting

1. (P1) Ranking the various generated paraphrase candidates
   - Dataset: QQP, Medical
   - Step1) Source sentence → Paraphrased sentence
   - Step2) Ranking the quality of paraphrased sentences with an automatic evaluation metric

2. (P2) Check the effectiveness of our approach on downstream tasks
   - Dataset: Financial, Hate Speech (eng), Hate Speech (kor)
   - To limit data-poor scenarios, we use randomly sampled balanced training data.
   - Step1) Data augmentation for training sentences
   - Step2) Comparison of model performance differences according to the presence or absence of augmented data

Approach

- Backbone
  - Pre-trained translation model: M2M100
  - M2M100 is a multilingual encoder-decoder model that can handle 100 languages

- Generation Framework
  1. (F1) Source-Encoder + Source-Decoder
     - (Ex. English) Sentence → English-encoder → English-decoder → New sentence
     - A kind of autoencoder
  2. (F2) Round-trip translation
     - (Ex. English&Korean) Sentence → English-encoder → Korean decoder → Korean-encoder → English-decoder
     - We used English, Korean, French, Japanese, Chinese, German, and Spanish as the language pool.
  - ➢ Decoder Options
     - (F1) Beam search with the beam size of 10 is used and the top-5 candidate sentences are generated
     - (F2) 3-beam-search is used in both the forward and backward paths, and the top-1 candidate sentence is generated
     - (Both) Do not overlap more than half of the length of the source sentence in succession with the source tokens
     - (Both) Prevented from generating repetitive 3-grams within the output sentence

- Ranking and Filtering
  1. Overlapping filtering
     - Remove sentences with only differences in case and space
  2. Diversity filtering
     - Score metric: Word Error Rate (WER) refers to the Levenshtein distance between the source sentence and the candidates
     - Only min(5, #num(overlap_cands)/2) sentences with a high diversity score are left
  3. Fluency filtering
     - Score metric: PPL (perplexity) using a language model (GPT2-medium)
     - Only min(3, #num(diversity_cands)/2) sentences with a low PPL
  4. Semantic filtering
     - Score metric: BERTScore leverages the contextual embeddings and matches words in the candidates and the source sentence by cosine similarity.
     - The candidate with the highest semantic score is chosen as the final sentence

Experiments

1. P1
   - Evaluation Metrics: Use different metrics than the ranking section
   - Semantic: Bleurt
   - Diversity: isacrebleu (= 100-sacrebleu)
   - Fluency: GPT2-small

Methods		QQP			Medical		
		Semantic	Diversity	Fluency	Semantic	Diversity	Fluency
		Bleurt	isacrebleu	PPL	Bleurt	isacrebleu	PPL
supervised	Edlp	-1.066	**86.843**	585.384	-	-	-
	Edlps	-0.857	83.504	597.024	-	-	-
unsuperivsed	UPSA	-0.729	65.749	392.833	-1.351	**89.418**	476.069
	CGMH(50)	-0.842	65.35	556.163	-1.405	88.95	818.307
	M2M100	0.036	43.539	346.17	-0.561	35.688	296.672
	Ours	**0.083**	69.421	**171.61**	**-0.508**	68.735	**158.76**
source	input sentence	0.124	0	270.781	-0.523	0	249.107
	gold reference	1	72.002	278.163	1	88.632	171.786

2. P2
   - Downstream task: classification task
   - Training model: BERT-base, Transformer
   - Data augmentation: M2M, Ours

Methods	augmentation	Financial	Hate Speech (eng)	Hate Speech (kor)
BERT-base	x	95.3	64.94	52.78
	M2M	95.15	66.2	54.52
	Ours	**96.33**	**68.31**	**55.03**
Transformer	x	80.47	53.24	**52.27**
	M2M	85.9	55.69	49.26
	Ours	**86.49**	**63.14**	51.04

Conclusion

- Our approach avoids the risk of relying on one model and one decoding option.
- However, our approach may suffer from speed issues for inferencing heavy models in parallel on one server.
- For real-service, it will be effective to extract candidates along with a simple model.
- In addition, our method can be used as data augmentation in data-poor environments.