Improved Prompt Generation for Knowledge Retrieval from Language Models

Shalini Roy
shaliniroy2022
@u.northwestern.edu

Vikram Kharvi
vikramkharvi2022
@u.northwestern.edu

Ethan Kohrt
ethankohrt2022
@u.northwestern.edu

Abstract

Knowledge can be retrieved from a language model by crafting a prompt and noting the model's prediction. Prompts can be written manually, or generated automatically in a variety of ways, such as by paraphrasing a manually-written prompt into many variations. We build on previous work in automatic prompt generation by exploring the impact of transformer-based paraphrasing techniques on prompt quality and knowledge retrieval.

1 Introduction

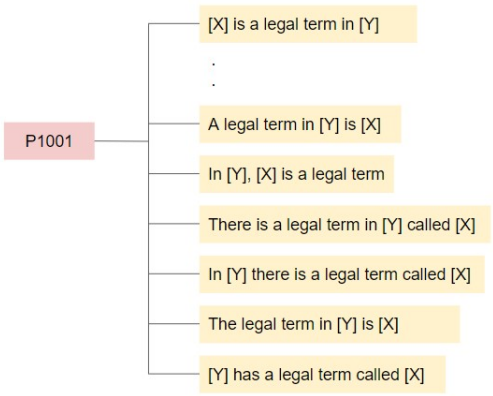
Recently, "How Can We Know What Language Models Know" (Jiang et al., 2019) explored automatic prompt generation techniques for extracting knowledge from language models. They experimented with mining prompts from Wikipedia text and generating prompts via paraphrasing. Specifically, the paraphrasing method they used was back-translation, where a prompt was first translated into another language and then back to English so that the candidate English translations could be used used as new prompts with the same meaning. We hypothesize that transformer-based paraphrasing techniques will produce higher-quality prompts, thus improving knowledge retrieval.

With the aim of developing improved prompt generation techniques for extracting knowledge relations from language models, we experiment with transformer-based paraphrasing techniques to expand manually-written prompts (Figure 1.) into many candidates prompts, as compared to the back-translation technique performed by Jiang et al. (2019).

1.1 Recent Works

Language models such as ELMo (Sarzynska-Wawer et al., 2021) and BERT (Devlin et al., 2018) have been shown to be very useful for a variety of common NLP tasks due to their capacity for linguistic knowledge and knowledge transfer. [(Peters et al., 2018), (Tenney et al., 2019)]. Their ability to generate text often demonstrates a high amount of latent world knowledge, though this knowledge is unstructured and only accessible via natural language prompts.

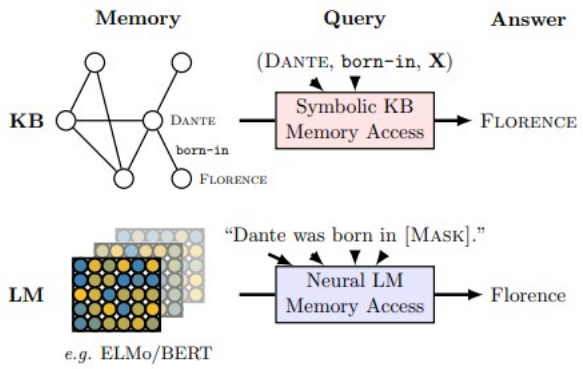
Figure 1: Examples of prompts generated by back-translation, One of the 41 relations, *P1001* expanded into multiple prompts by back-translation.



Knowledge Bases, on the other hand, are the ideal for storing gold-standard or factual relational data, which can be accessed through a query language. They are often constructed manually or automatically extracted from natural language, so it would be convenient if existing pre-trained language models could be made to function as knowledge bases. (Figure 2.)

(Petroni et al., 2019) formulated a method to extract factual knowledge from natural language in the form of *triplets* $< x, r, y >$, where x is the subject, y is the object, and r is the relation between the subject and the object. The 'LAMA' benchmark introduced by this work formed the basis for 'LPAQA,' which explored techniques for prompt generation such as mining prompts from Wikipedia and paraphrasing via back-translation (Jiang et al.,

Figure 2: Querying knowledge bases (KB) and language models (LM) for factual knowledge. Jiang et al. (2019)



2019). We build on this work by exploring the effect of transformer-based paraphrasing on the LAMA knowledge retrieval benchmark.

2 Model

For paraphrasing, we use the Hugging Face implementation of 'Pegasus,' a language model fine-tuned for paraphrasing (Zhang et al., 2020). For our knowledge-retrieval experiments, we employ the 'fill-mask' pipeline from Hugging Face, and we compare results from BERT-base and a BERT-base model fine-tuned on masked example prompts.

2.1 Paraphrasing-based Prompt Generation

We want to generate prompts $\{t_{r,i}\}_{i=1}^{T}$ for each of the 41 relations r in the T-REx subset of LAMA. The T-REx subset contains a manually-written prompt template for each relation, such as "[X] is the capital of [Y]", which we input directly into the Pegasus paraphrasing model to produce variations, such as "[Y] has [X] as its capital" where [X] is the subject and [Y] is a single-word object to predict. We instruct the paraphraser to generate 40 paraphrases with a max length of 20 tokens, then filter out those that no longer contain exactly one [X] and [Y]. When querying the language model, we replace [Y] with [MASK] and replace [X] with the subject:

The field of *[MASK]* is what Silva works in.
obj_label sub_label

2.2 Fine-tuning on BERT$_{base}$

We try to explore the BERT$_{base}$ model's performance on our particular T-REx dataset by using MLM as our fine-tuning objective. Our intuition is that if our model learns about the structure of subject-object pairs then the object label predictions will improve.

3 Datasets

3.0.1 T-REx:

Following Jiang et al. (2019), we use the T-REx subset of the LAMA dataset, which consists of 1,304,391 knowledge triples for 41 relations. Each example in the dataset contains a subject-object pair that satisfies some relation, and a text template that we use to generate query prompts. We import the dataset through the hugging face 'datasets' library [1].

We use the following fields from T-REx subset:

- obj_label: A string label for the object slot
- sub_label: A stringlabel for the subject slot
- predicate_id: The predicate/relationship
- template: A pattern of text for extracting the relationship, object and subject of the form "[X] some text [Y]", where [X] and [Y] are the subject and object slots respectively

3.0.2 T-REx-train:

We also borrow the training dataset constructed by Jiang et al., called T-REx-train, which is used for evaluating the quality of generated prompts. It was constructed to contain 1000 examples for each of the 41 relations in T-REx, but without any overlap with T-REx. The authors provide this dataset here: https://github.com/jzbjyb/LPAQA.

4 Experiments

4.0.1 Ranking Prompt Templates:

To rank the quality of our prompt templates, we test each template's accuracy on predicting objects in the T-REx-train set. We sample a subset of 100 subject-object pairs to reduce compute time, construct queries by combining training examples with the template, and compare the predictions of the BERT-base unmasker to the ground-truth object. BERT-base performed somewhat better than the fine-tuned model, so we use the scores from BERT-base (Figure 5).

[1] the original dataset can also be found at https://dl.fbaipublicfiles.com/LAMA/data.zip

Figure 3: BERT-base attains higher accuracy on the training set than the fine-tuned model on most relations.

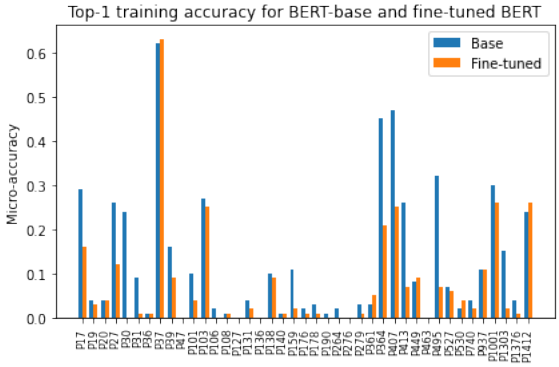


Figure 4: BERT-base attains higher accuracy on the testing set than the fine-tuned model on most relations.

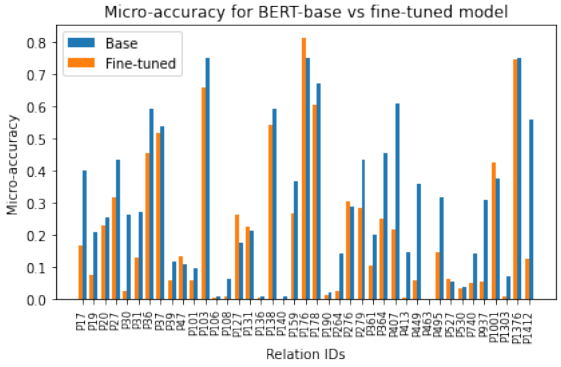
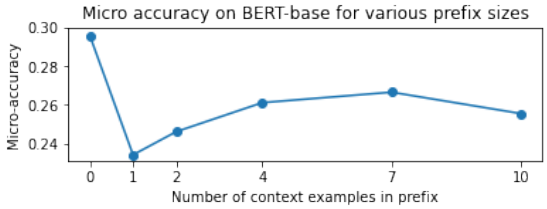


Figure 5: Using training examples as a prefix leads to a decrease in accuracy.



4.1 Accuracy: Micro and Macro

Standard accuracy (micro-averaged accuracy) fails to account for the fact that the data in the T-REx subset is imbalanced; some objects are over-represented in the data, so there is potential for the micro-accuracy score to be artificially inflated. Simply choosing the majority object in each relation can achieve a score as high as 22.0%, while the stricter metric of macro-averaged accuracy reduces this score to 2.2% (Jiang et al., 2019). To calculate macro-averaged accuracy for a relation, first calculate accuracy for each unique object separately, then average those to get the relation-level accuracy. We report both metrics for consistency with previous works.

4.2 Methods

We compare accuracy scores from probing standard BERT-base and our fine-tuned BERT-base model. Only the top-scoring prompt template was used for each relation. We test both models on the subject-object pairs in the LAMA T-REx subset, after removing duplicate pairs, measuring micro- and macro-accuracy. To add additional context to the query, we experimented with adding a prefix to the start of the prompt consisting of some number of random true examples from the training set; so for a prefix size of two, the original query "[MASK] is the capital of France" might be modified to "Cairo is the capitol of Egypt. Buenos Aires is the capitol of Argentina. [MASK] is the capitol of France." We also report the effect of filtering NLTK stopwords and single punctuation tokens from the language model outputs.

5 Results

5.1 Base vs Fine-tuned

In Table 2. we see the micro-averaged and macro-averaged accuracy for the BERT-base model and the fine-tuned BERT-base model. The fine-tuned model performed worse than the base model, achieving a micro-averaged accuracy score of 0.20 compared to the base-model's score of 0.29.

5.2 Prompt Engineering

Concatenating randomly sampled training examples to the beginning of the prompt seems to decrease accuracy significantly. As shown in Figure 5, concatenating one example to the prompt decreases the micro-accuracy from 0.29 to 0.23. Longer prefixes performed better up to a point, but still did not meet the accuracy with no prefix.

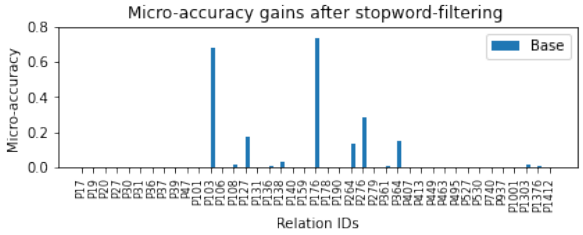
Model	Micro	Macro
LPAQA$_{\text{Best}}$	0.396	0.439
LPAQA$_{\text{Man+Para}}$	0.341	0.228
Model$_{\text{base}}$	0.29	0.22

Table 1: Comparing results to Jiang et al.

Model	Micro	Macro
Model$_{\textbf{base}}$	0.29	0.22
Model$_{\textbf{fine-tuned}}$	0.20	0.19

Table 2: BERT-base vs fine-tuned BERT-base

Figure 6: Some relations see major accuracy gains after stopword-filtering.



5.3 Stopword Filtering

Filtering stopwords from the language model predictions led to an overall increase in micro-accuracy on BERT-base from 0.24 to 0.29. Overall macro-accuracy similarly increased from 0.18 to 0.22. The improvement was not evenly distributed across relations; most saw no improvement, while others saw major micro-accuracy improvements as high as +0.74 (Figure 6).

6 Analysis

Using the transformer-based Pegasus paraphraser did not result in improved scores over Jiang et al. Table 1 compares their scores using back-translation paraphrasing (**LPAQA$_{\textbf{Man+Para}}$**), their highest score on BERT-base (**LPAQA$_{\textbf{Best}}$**), and our technique's best performance on BERT-base (**Model$_{\textbf{base}}$**). One hypothesis that partially explains this shortfall is that our prompt generation missed certain successful prompts. For instance, Jiang et al. identified that the manually-written template for the relation *subclass_of* could be improved 22.7% by re-wording "[X] is a subclass of [Y]" to "[X] is a type of [Y]." None of our generated prompts even contained the word 'type.' This issue might be addressed by increasing the number of generated paraphrases while decreasing the max token length to force the paraphraser to use synonyms rather than shuffling word order.

Concatenating training examples to the query did not increase performance. One possible explanation is that the subjects and objects in the training set may be more obscure than those in the testing set. Comparing Figures 3 and 4, we see that accuracy on the training set was much worse than on the testing set. We hoped that adding examples to the prompt as context would cue the model to predict similar kinds of words, but if it had little knowledge about the entities in the example, then the context might just be noise.

Filtering stopwords did lead to major accuracy gains for some relations. We strongly suspect that performance could be further improved with additional constraints, such as filtering for correct part of speech.

7 Future Works and Conclusion

In this paper, we explored abstract summarization as a paraphrasing technique to generate paraphrases for prompts. We compared the performances of BERT$_{base}$ and BERT$_{fine-tuned}$ model against the LPAQA models. We employed some prompt-engineering and stopword filtering, and noticed that stopword filtering gained a very significant micro-accuracy improvement on some of the prompts.

One interesting observation was that our fine-tuning model did not perform as well as the base model. For future works, more effective fine-tuning can be explored.

—

References

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*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2019. How can we know what language models know? *CoRR*, abs/1911.12543.

Matthew E Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. *arXiv preprint arXiv:1808.08949*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. 2021. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304:114135.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.