Data-Efficient Concept Extraction from Pre-trained Language Models for Commonsense Explanation Generation

Yanbo Fang
Department of Computer Science
Rutgers University
New Brunswick, NJ, US
yanbo.fang@rutgers.edu

Yongfeng Zhang
Department of Computer Science
Rutgers University
New Brunswick, NJ, US
yongfeng.zhang@rutgers.edu

Abstract

Predicting the key explanation concept is essential for generating commonsense explanations. This paper introduces a method to predict the concept from pre-trained language models for commonsense explanation generation. Our experiment found that adopting a language model as the concept extractor and fine-tuning it with 20% training data can improve the quality and accuracy of the generated explanations over multiple evaluation metrics. Compared with conventional methods that search concepts over knowledge graphs, our method does not require the preparation and training models to search through knowledge graphs. To better understand the results from pre-trained language models, we also designed a metric to evaluate the retrieved concepts. Through analysis and experiments, we show the correlation between this metric and the performance of the generators, and we also show the importance of attaching concepts for generating high-quality sentences.

1 Introduction

The natural language processing (NLP) field has gigantic advancements since the involvement of deep learning (Young et al., 2018) and large-size pre-training models (Liu et al., 2019; Devlin et al., 2019; Radford and Narasimhan, 2018). However, creating human-like AI systems requires models comprehending commonsense knowledge and reasoning, which is still a challenge for current NLP models (Jia and Liang, 2017). Among all the research about making language models' responses fit common sense knowledge, we would like to focus on generating sentences to explain a statement with commonsense knowledge (Wang et al., 2019). An example of this task is shown in Table 1. Given a statement, the goal is to output an explanation to explain the statement.

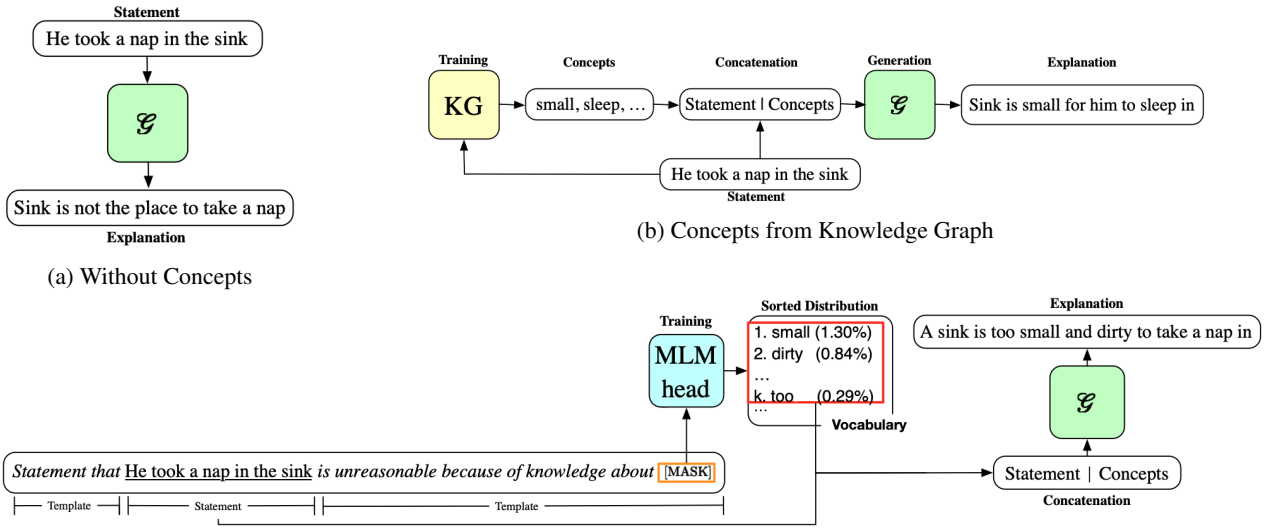
There are two ways for this task at present. One way, Figure 1(a), is to let language models to generate an explanation directly given one statement without external assistance (Wang et al., 2019, 2020). The benefit is that this is an end-to-end pipeline and would be easy to use, but it would be hard to interpret why models output such sentences, and most times, the performance would be limited. Another method, Figure 1(b), is extracting bridge concepts from external structured knowledge graphs to connect statements and explanations (Ji et al., 2020). With the assistance of knowledge graphs, such as ConceptNet (Liu and Singh, 2004), this method can introduce ground truth knowledge to offer helpful information for generation, and the selected concepts can be used to interpret models' outputs. Our experiments in Figure 3 confirm that informative concepts can assist the generation process. However, this method would need additional training to search concepts, and this method would be limited under a constrained environment where structured data would be rare.

This paper introduces a pre-trained language model driven method, Figure 1(c), to combine the advantages of both methods. The key is that we can adopt

Statement: The school was open for summer
Template: Statement that *the school was open for summer* is wrong because concepts about `[MASK]` .
Concepts: summer, school, education, the, schools, time, it, vacation, holidays, students, learning, kindergarten, that, teaching, children
Explanation: Summer time is typically vacation time for school.

Table 1: Example of the task and our method. The task is to produce an explanation for a given statement. Our method adopts masked language modeling to firstly retrieve concepts from pre-trained language models by transferring statements into templates before generation.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 5912 - 5922



(a) Without Concepts

(b) Concepts from Knowledge Graph

(c) Concepts from pre-trained Language Model

Figure 1: This Figure gives illustrations of (a) the standard method for generations without concepts augmentation, (b) the method enhanced by concepts from knowledge graphs and trained models, and (c) the method enhanced by k concepts from pre-trained language models without training for this explanation generation task. KG is a knowledge graph. $\mathcal{G}$ is a generator model. PLM means a pre-trained language model.

pre-trained language models to provide bridge concepts through masked language modeling (Devlin et al., 2019) and prompt-tuning (Liu et al., 2021). Our approach is composed of two distinct models, *conceptor* and *generator*. The conceptor is used to provide bridge concepts for a given statement. The generator is responsible for generating explanations based on the combining statement and concepts. Following prompt-based tuning (Schick and Schütze, 2021; Gao et al., 2021; Raffel et al., 2020), we design a template to transfer a generation question to a masked prediction task. Then we use data-efficient finetuning to modify pre-trained language models such as BERT or RoBERTa to encode statements, and we pick up candidates with the highest scores in vocabulary as concepts by masked language modeling. An example is in Table 1. In the next step, the selected concepts are attached at the end of the statements as the generator's input to generate explanations. This method arises with two motivations. One is that pre-trained language models could implicitly encode world knowledge (Niven and Kao, 2019; Talmor et al., 2020; Davison et al., 2019). Another is that while pre-trained models find it challenging to produce sentences that call for world knowledge, it would be simpler for them to output key concepts without taking syntax and grammar into account (Petroni et al., 2019).

We discovered that concepts from pre-trained language models enhance the metric scores of generated sentences compared to baseline models. Compared with structured knowledge graphs, pre-trained language models are widely developed and convenient for processing unstructured texts. Moreover, fine-tuning the conceptor with more data points would further improve the efficiency of concept retrievals and metric scores. Furthermore, we design experiments to identify the advantages of obtaining informative concepts from pre-trained language models for explanation generations.

Our contributions can be summarized as follows:

- We proposed to replace annotated knowledge graphs with pre-trained language models for concept extraction. Our data-efficient training method demonstrates its advantages in the explanation generation tasks.
- We designed a metric to assess the retrieved concepts and showed the association between the metric scores and created sentence quality.
- We analyzed the challenges of retrieving concepts from current pre-trained language models and how these concepts would help generate better explanations.

2 Related Work

Knowledge of Pre-trained Language Model. Pre-trained language models are now the mainstream of the Natural Language Processing community and demonstrate incredible strength in solving complex tasks, such as text understanding, generation, and domain generalization (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020; Hendrycks et al., 2020). However, these models make inferences in a human-agnostic way and are easy to be attacked by attaching a trigger (Wallace et al., 2019). Therefore, some researchers are trying to interpret what these models have learned in their pre-training. For example, (Hewitt and Manning, 2019) shows it can recover syntactic dependency from token representations from BERT, and pre-trained language models contain a sense of world knowledge (Petroni et al., 2019; Li et al., 2021a).

Moreover, to reduce the gap between pre-training and fine-tuning, the prompt-based method is proven an efficient way to improve pre-trained language models' performance on downstream tasks (Gao et al., 2021; Schick and Schütze, 2021; Le Scao and Rush, 2021). Thus, we adopt prompt-tuning and designed templates to express pre-trained language models' internal knowledge to replace the external knowledge graphs.

Reasonable Explanation Sentence Generation. Some of the current works (Ji et al., 2020; Lin et al., 2020) focus on adjusting models to output sentences that fit the context and commonsense knowledge. Because investigations (Zhou et al., 2020; Richardson and Sabharwal, 2020) find that current pre-trained models have disadvantages in solving tasks needing inferences steps, some of these research (Moon et al., 2019; Zhou et al., 2018; Guan et al., 2019) propose using a knowledge graph to provide additional concepts to ensure controllable generation.

Natural language explanation generation is also extensively explored in explainable recommender system research (Zhang et al., 2020), which aims to generate personalized and reasonable natural language sentences to explain why certain items are recommended to users. Early methods used manual templates to generate explanation sentences (Zhang et al., 2014), and more recent works explored neural-template generation methods (Li et al., 2020a,b), personalized transformers (Li et al., 2021c, 2022b), visual-enhanced explanation generation (Geng et al., 2022a; Chen et al., 2019), explanation ranking methods (Li et al., 2022a, 2021b), counterfactual explanations (Tan et al., 2021, 2022), logical explanations (Shi et al., 2020; Chen et al., 2021, 2022a,b; Zhu et al., 2021), path-based explanations (Geng et al., 2022b; Xian et al., 2019, 2020), as well as large language models for explanation generation such as the P5 large recommendation model (Geng et al., 2022c). Following previous work, we use pre-trained language models as a knowledge base to replace annotated knowledge graphs so as to provide concepts for commonsense explanation sentence generation.

3 Method

3.1 Problem Setup

In this task, conditioning on an statement $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ with length m, where x_i denotes the ith tokens, a model is expected to generate a sentence $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ with length n through model $f : \mathbf{x} \rightarrow \mathbf{y}$ to explain why the input statement is against common sense.

3.2 Structure Overview

In our method, we adopt two separate pre-trained language models, one is for concept extractions at the first stage, and we call this model as *conceptor* and denote as $\mathcal{C}$. *Generator*, denoted as $\mathcal{G}$, is used to generate explanations based on extracted concepts $\mathbf{c}$ and the original statement $\mathbf{x}$. The whole process can be formulated as

$$P(\mathbf{y}, \mathbf{c} \mid \mathbf{x}) = \underbrace{P(\mathbf{c} \mid \mathcal{M}(\mathbf{x}), \theta_{\mathcal{C}})}_{\text{Conceptor}} \cdot \underbrace{P(\mathbf{y} \mid \mathbf{x}, \mathbf{c}, \theta_{\mathcal{G}})}_{\text{Generator}} \tag{1}$$

3.3 Concepts Extraction

This part is the $P(\mathbf{c} \mid \mathbf{x}, \theta_{\mathcal{C}})$ in equation 1. Firstly we modify inputs $\mathbf{x}$ by attaching a template to create a masked sentence $\mathcal{M}(\mathbf{x})$. The template we use is the template with Id 0 in Table 3 and an example is in Table 1. Then we use $\mathcal{C}$ to encode $\mathcal{M}(\mathbf{x})$. At this step, we can obtain the representation of the `[MASK]` tokens and send it to a prediction head $\mathbf{W} \in \mathbb{R}^{d \times N}$, where d is the representation size and N is the vocabulary size, to produce a probability of a word fits into the `[MASK]`. Subsequently, we select k candidates $\mathbf{c} = \{c_1, c_2, \ldots, c_k\}$ with the highest probability. k is a hyperparameter, and the default setting is 3. Concepts $\mathbf{c}$ combined with

input sentence $\mathbf{x}$ are sent into model $\mathcal{G}$ for explanation generation. In the default setting, we only train our conceptor with 20% training data in a weak-supervision manner. We take the tokens in the ground truth explanations but not seen in the input statements as the label. The detail about fine-tuning the model is described in section 5.4.

3.4 Explanation Generation

This section will describe the $P(\mathbf{y} \mid \mathbf{x}, \mathbf{c}, \theta_{\mathcal{G}})$ process in equation 1. The concepts $\mathbf{c}$ from previous step are concatenated at the end of the inputs $\mathbf{x}$ to construct a new input $(\mathbf{x}, \mathbf{c}) = (x_1, x_2, \ldots, x_m, c_1, c_2, \ldots, c_k)$, and feed the model $\mathcal{G}$ with the new input. The generator model follows the equation 2 to produce an explanation.

$$P(\mathbf{y} \mid \mathbf{x}, \mathbf{c}, \theta_{\mathcal{G}}) = \prod_{t=1}^{n} P(y_t \mid \mathbf{x}, \mathbf{c}, \mathbf{y}_{<t}, \theta_{\mathcal{G}}) \quad (2)$$

The structure of the $\mathcal{G}$ would be a sequence-to-sequence model, which contains an encoder and decoder. The Encoder would map inputs $(\mathbf{x}, \mathbf{c})$ to a high-dimensional contextualized representations. The decoder would condition input representations and past-generated sentences to generate the following sentences.

The generator $\mathcal{G}$ would optimize its parameters towards minimizing the following loss function:

$$\mathcal{L} = -\log P(\mathbf{y} \mid \mathbf{x}, \mathbf{c}, \theta_{\mathcal{G}}) \quad (3)$$

4 Experiments

4.1 Data

Similar to earlier work (Ji et al., 2020), we also evaluate the model on the dataset from Commonsense Validation and Explanation Competition. This challenge tests the commonsense knowledge through three subtasks: (1) selecting the statement with the proper commonsense knowledge from two similar options. (2) Choose the reason from three options to answer why a statement is against commonsense. (3) Coming up with explanations for why a specific statement is absurd. In this work, we focus on task 3, the explanation generation task.

This dataset collects $10,000$ counter-commonsense claims, paired with three human-written explanations for each. We split each training data into three statement-explanation pairs. The final dataset is made up of $24,000(80\%)$ training data, $1,000(10\%)$ development data and $1,000(10\%)$ testing data.

4.2 Model

We adopt various pre-trained language models as the concept extractor. Models we used are BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), with both base and large versions. Then we employ the T5-small (Raffel et al., 2020) model as the generator that accepts the combination of statements and concepts as inputs and outputs the explanation. The resulting explanations are evaluated using the automatic metrics, BLEU (Papineni et al., 2002), ROUGE (Lin, 2004) and METEOR (Banerjee and Lavie, 2005).

4.3 Implementation Details

At the start, We trained the conceptor with 20% training data, then we used the trained conceptor to process all the data in training, development, and testing sets to produce concepts for the generator.

The default hyperparameter setting is following: number of concepts selected $k = 3$, template is id 0 in Table 3, batch size is 8 for conceptor and 32 for generator training, training epoch is 5 for conceptor and 3 for generator, learning rate is $2e-5$ for conceptor and $1e-3$ for generator. The model with the best BLEU-4 score during evaluation would be saved and evaluated on test data.

4.4 Main Results

Table 2 is our main results. Compared with the generator-only method (T5-small) and knowledge graph augmentation method, using a pre-trained language model as the extractor can improve most scores and achieve the best results. Another observation is that large-version models typically outperform their base version. We attribute the reason that larger models tend to be more powerful (Liu et al., 2019; Hendrycks et al., 2020; Desai and Durrett, 2020; Brown et al., 2020). More analysis is located in section 5.4. Moreover, we can see that attaching concepts chosen randomly (Random) would reduce the result, indicating that only attaching appropriate concepts would best aid the generation process.

Additionally, to investigate the profit of concepts, we attached the tokens exclusive in references at the end of the statement and transmitted the combination to the generator. Consequently, we can observe a significant boost by attaching the correct

Models	ROUGE-1/2/L/Lsum	METEOR	BLEU-3	BLEU-4
T5-small	37.18/12.81/33.21/33.19	29.70	28.09	18.51
w/ Random concepts	34.15/11.64/29.93/29.16	27.37	26.44	16.37
w/ Knowledge Graph	37.01/12.60/33.23/33.21	29.96	28.36	19.33
w/ BERT-base	37.10/12.89/33.18/33.24	29.74	28.73	19.67
w/ BERT-large	37.20/**13.52/33.52/33.51**	30.02	**30.10**	**21.04**
w/ RoBERTa-base	37.15/13.38/33.25/33.20	**30.39**	29.39	20.10
w/ RoBERTa-large	**37.33**/13.26/33.38/33.37	30.34	29.62	20.23
Upper Bound	43.75/25.74/38.26/38.31	-	47.93	42.55

Table 2: Metric evaluation results for explanations from different methods. All these methods use T5-small as the generator. The bold number indicates the best results. **T5-small**: Using T5 to generate explanations without any concept augmentation. **Random**: Attaching concepts selected randomly. **Knowledge Graph**: Following the method in (Wang et al., 2019) that looking up concepts from ConceptNet. The model searching through ConceptNet is trained with 100% data. **BERT-base/large** and **Roberta-base/large** denote our method and the model we used as conceptor. The conceptor is trained with 20% data. **UpperBound**: Attaching the concepts that appear in references but do not exist in input statements. Therefore, we consider this result from **UpperBound** to be the upper limit of our method with selected models and trained data size.

concepts. Please note, we define the "correct" of concepts as the tokens are not seen in the statement but are included in corresponding references at this place. More discussion about retrieved concepts is in the section 5.5. This demonstrates how enhancing retrieving concept precision from language models would be beneficial for explanation generation. We have a deeper discussion in the section 5.5.

5 Analysis

5.1 Effects of Different Templates

Since template engineering is an important part of our method, we are curious to see whether these pre-trained models are sensitive to the design of templates. Table 3 shows the model's performance with different manually created templates for prompt-tuning. From this table, although there are some variations of the BLEU-4 score, the standard deviation is only about 0.1259. Therefore, the template choice has a small impact on the metric score, indicating that pre-trained models are resistant to the difference in templates.

Moreover, we conduct additional ablation experiments. Results are in Table 4. We study what if adopting a traditional fine-tuning method that takes the `[CLS]` representation for concept extraction rather than adding a template. Even though models tend to promote under few-shot learning with prompt-based tuning (Gao et al., 2021), there is a small-scale decrease in our result. The reason could be a large amount of data would fill the gap between traditional fine-tuning and prompt-tuning. Next, we study the effects of the ensemble of templates with the formula 4 and randomly selected templates with the formula 5.

$$P(\mathbf{c} \mid \mathbf{x}, \theta_{\mathcal{G}}) = \frac{1}{M} \sum_{i}^{M} P(\mathbf{c} \mid \mathcal{M}_i(\mathbf{x}), \theta_{\mathcal{G}}) \quad (4)$$

$$P(\mathbf{c} \mid \mathbf{x}, \theta_{\mathcal{G}}) = P(\mathbf{c} \mid \mathcal{M}_i(\mathbf{x}), \theta_{\mathcal{G}}), 1 \leq i \leq M \quad (5)$$

M is the number of distinct templates, and i denotes the Id of templates in 3. In formula 5, i is uniform randomly selected from the range $[1, 2, \ldots, M]$ for each training data. Table 4 shows these two methods can give enhancement compared to the default setting, and all single template results are in Table 3. Therefore, one advantage of prompt-tuning data augmentations with templates.

5.2 The Number Of Attached Concepts

To further investigate the best number of retrieved concepts for models, we designed an experiment by gradually changing the hyperparameter k. k denotes the number of tokens selected from masked language modelings and attached with statements.

The results are shown in Figure 2. We use BLEU-3 and BLEU-4 to evaluate the model's outputs, and the pre-trained models selected for retrieving concepts are the base version of BERT. The x-axis

Id	Template	BLEU-4
0	Statement that [DATA] is unreasonable because of knowledge about [MASK]	19.67
1	knowledge about [MASK] can refute that [DATA]	19.72
2	[DATA] is wrong because bridge concept of [MASK]	19.84
3	[DATA] can be explained by knowledge about [MASK]	19.58
4	knowledge about [MASK] disproves statement that [DATA]	19.89

Table 3: Templates used in our method. [DATA] would be replaced by the original data. The template is created manually and has semantic meanings.

Model	BLEU-4	METEOR
Default Setting	19.67	29.74
w/o Template	19.58	29.66
w/ Ensemble	20.21	30.03
w/ Random	20.28	30.14

Table 4: Template Ablation Study with BERT-base as conceptor

denotes the number of concepts k, and the y-axis represents the evaluation scores on test data.

In this figure, the model provides the best scores overall metrics with $k = 3$. For k small than 3, the performance is proportional to k. However, when k exceeds 3, the results decrease as k increases. We argue that although our method can bring some concepts that would benefit the explanation generation stage, it also would inevitably include some noisy tokens that would confuse the generation model. As long k is less than 3, the benefit from retrieved concepts outperforms the noise. However, when the k becomes larger than k, the noise would reduce the advantage of these concepts, thus decreasing the scores. Therefore, in our future work, the key to enhancing the method would be raising the accuracy of concept retrieval and reducing the noise.

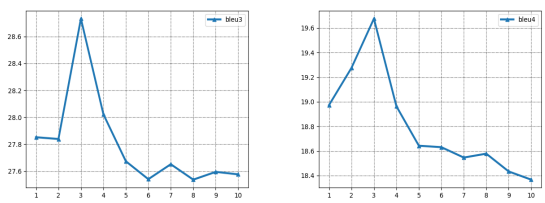


Figure 2: Performance by changing the number of top-k concepts selected.

5.3 Evaluating Retrieved Concepts

In this section, we would like to talk about how to evaluate retrieved concepts and introduce an automatic evaluation metric. We also provide human analysis in section 5.6.

Inspired by the concept F1 metric from (Ji et al., 2020) to evaluate sentences, tokens that are in the grounded truth explanations but do not appear in the input sentences are considered the positive class. All other tokens in the vocabulary are regarded as the negative class.

In detail, we assume all unique tokens in tokenized input x as C_{x}, unique tokens in tokenized ground truth explanation y as C_{y}, and extracted concepts $\hat{\mathrm{y}}$ from conceptor as $C_{\hat{\mathrm{y}}}$. $C_{\mathrm{y}} - C_{\mathrm{x}}$ represents the desired concepts. Then we use precision to measure the match conditions among references and outputs.

$$\text{Precision} = \frac{|C_{\hat{\mathrm{y}}} \cap (C_{\mathrm{y}} - C_{\mathrm{x}})|}{|C_{\hat{\mathrm{y}}}|} \tag{6}$$

$C_{\hat{\mathrm{y}}} \cap (C_{\mathrm{y}} - C_{\mathrm{x}})$ are overlapped concepts, and $|\cdot|$ denotes the number, $|C_{\hat{\mathrm{y}}}|$ is usually the same as k. This metric result shows how many retrieved concepts are aligned with the grounded truth explanations. We choose not to use recall, equation 7, for evaluation as we think the recall in our experiment setting can not properly measure the retrieved concepts and ground truth.

C_{y} contains many ambiguous and irrelevant concepts, which are hard to retrieve, such as *do, does, not, is, are, a, an*. However, $|C\ mathrm\hat{y} \cap (C_{\mathrm{y}} - C_{\mathrm{x}})|$ are less or equal to k, but $|C_{\mathrm{y}}|$ is usually much larger than k.[1] All these factors make the recall score always low regardless of the quality of retrieved concepts and thus not a good indicator.

[1]The average number of concepts in each data point is 7.26.

Models	ROUGE-1/2/L/Lsum	METEOR	BLEU-3	BLEU-4
BERT-base				
0%	37.15/12.84/33.11/33.13	29.31	28.38	19.16
20%	37.10/12.89/33.18/33.24	29.74	28.73	19.67
50%	37.30/13.07/33.80/33.75	29.71	28.86	19.83
100%	37.50/13.15/33.91/33.89	29.84	29.61	20.47
RoBERTa-large				
0%	37.09/12.77/33.16/33.18	29.78	28.30	18.60
20%	37.33/13.16/33.38/33.37	30.24	29.62	20.23
50%	37.58/13.63/33.62/33.68	30.28	30.00	20.78
100%	37.84/13.88/34.97/33.95	30.46	30.69	21.72

Table 5: Consequences after fine-tuning conceptors with $k = 3$. Models represent different concept extractors. % denotes the percent of data used for training.

Data	1	3	5	10	20
BERT-base					
0%	2.40	1.87	2.16	2.36	2.08
20%	20.00	16.16	13.02	9.55	6.71
100%	30.60	22.20	17.84	12.61	8.59
RoBERTa-large					
0%	2.10	2.03	2.28	2.12	2.23
20%	38.20	26.73	21.18	14.78	9.84
100%	47.50	34.80	28.42	20.15	13.83

Table 6: The precision metrics scores for concepts from pre-trained language models with different top-k. The number before % means the percent of data used to train the conceptor. 0% is zero-shot, and 100% is fully fine-tuning. The higher, the better.

$$\text{Recall} = \frac{|C_{\hat{y}} \cap (C_y - C_x)|}{|C_y|} \tag{7}$$

Then we use the precision in equation 6 to evaluate the concepts collected from BERT-base and RoBERTa-large. The result is shown in Table 6. We can observe these scores are low when zero-shot, but the precision is much improved by fine-tuning with 20% data. Furthermore, larger models perform better, which make it consistent with the result in section 4.4 and section 5.4. However, even after training with 100% data, the best result that the model can reach is still 47.50 by Roberta-large. This indicates this task is still challenging for current models. Results decrease as k grows, indicating that bigger k tends to bring more noise and interrupt the generation.

5.4 Fine-tuning The Conceptor

In the last section, we observe that training with more data and larger models can increase the precision of concept retrieval, but we also wonder whether these more accurate concepts would benefit the final results. The same as the precision described in section 5.3, we set all the concepts in $(C_y - C_x)$ as desired concepts for masked language modeling and denote it as label C. Because this task can be regarded as multi-label learning, we adopt the sigmoid function to map the output scores into probability and keep each value independent. Then we optimize conceptors by cross-entropy loss.

As the results in Table 5, fine-tuning increases metric scores with the percent of data used. Comparing zero-shot and complete training, there is a 1.31 BLEU-4 increase for BERT and a 3.12 BLEU-4 increase for RoBERTa. Therefore, this evidence underlines the connections between the concept of precision metric and model performance. In addition, RoBERTa is better than BERT overall metrics trained with the same percent of data, the same as observations before.

5.5 How Retrieved Concepts Affect Generations

Our proposed method can simplify concept extraction and benefit the generation process more efficiently. However, current methods still face difficulty guiding pre-trained language models to return correct concepts as the scores in Table 6 are still deficient.

Model	Correct	Reasonable	Dependent	Relevant
T5-small	-	0.400	-	-
w/ KG	0.147	0.440	0.12	0.253
w/ BERT	0.180	0.520	0.133	0.293

Table 7: Human evaluation of generated explanations and selected concepts.

Therefore, we design experiments in which m of k concepts are fixed as correct and $k-m$ concepts as noisy tokens, where $1 \leq m \leq k$. Specifically, we randomly sample m concepts that appear in the ground truth explanations but are not included in the input sentence and pick up $m-k$ noisy concepts from the vocabulary at random. Then shuffling these k concepts followed by attaching these concepts at the tail of statements for generations. we ran experiments with $k \in 1, 3, 5, 8$ and m increases from 1 to k for each k, and evaluate with BLEU-3 and BLEU-4 metrics.

Figure 3 displays the results. We see that the score rises with the number of correct concepts increasing. Additionally, smaller k typically performs better when having the same m. Models only need an expectation of 2 to easily surpass our best results in Table 2. These observations suggest that enhancing models' capacity to identify better bridge concepts could be the focus of future work for generating commonsense sentences.

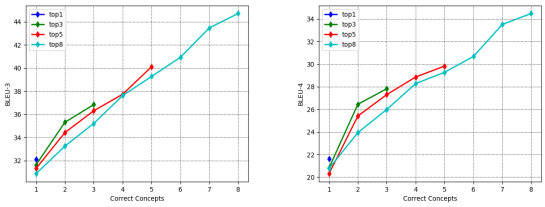


Figure 3: Performance by changing the number of expectations. The extractor is BERT-base, and the generator is T5-small.

5.6 Human Analysis

As these automatic evaluation metrics have the difficulty of heavily relying on the quality of human references, we also provide a human investigation to see whether our method can help generation compared with the baseline. We collect 50 statements paired with concepts, generated explanations, and grounded truth explanations. Then one human evaluates these data with 4 standards. **Correct**: whether retrieved concepts occur in the grounded truth explanations, whether these concepts are in generated explanations or not. **Reasonable**: whether generated explanations explain statements with commonsense knowledge. **Dependent**: How many concepts are used in generating concepts? **Relevant**: this critique measures the ratio of concepts related to the statement's topic but unseen in the ground-truth explanation.

Example1	
Statement:	mustard must go on all sandwiches.
T5-small:	mustard is not a condiment.
w/ KG:	mustard is not edible.
w/ BERT:	it is unnecessary to have mustard
Concepts KG:	sandwich cook edible
Concepts BERT:	people unnecessary food
Example2	
T5-small:	he was taking bath in the kitchen.
Baseline:	you can't take a bath.
w/ KG:	kitchens are not a place for bathing.
w/ BERT:	you take bath is bath room.
Concepts KG:	truth bath power
Concepts BERT:	bath meat room
Example3	
T5-small:	I want to learn how to play a dog
Baseline:	dogs are not for play
w/ KG:	you should not play a dog.
w/ BERT:	dogs are not instruments.
Concepts KG:	game learning animal
Concepts BERT:	instrument fun video

Table 8: Sample cases of extracted concepts and generated explanations. Red color means concepts appeared in generated explanations

The results is in Table 7 and Table 8 are some sample cases. Table 7 indicates that our method can select relevant concepts more effectively and generate more reasonable explanations compared with baselines. We also observe that **Correct** is always better than **Dependent**, which means that the generator still mainly relies upon their internal knowledge for generations.

Table 8 provides case examples in the test dataset

with produced explanations and concepts from our method and baseline models. Although in these examples, models can create explanations with the aid of concepts, these explanations follow a simple structure and grammar without further expansion.

6 Conclusion

In this paper, we present a combining prompt-tuning and pre-trained language models concept extraction method for commonsense explanation generation task. We demonstrate that our method is more data-efficient for concept extractions than methods that select concepts from the knowledge graph. Our method improves evaluation metric scores compared with baseline models. Additionally, we analyze the method of selecting concepts from pre-trained language models and challenges to solve in future work.

Limitation

In our experiment, we find no very efficient methods or pre-trained language models to retrieve concepts sufficiently. We believe this limitation would restrict the generator's capacity to generate explanations, and future work would focus on proposing methods to improve the concept retrieval precision. Additional limitations would be we haven't tested our method with alternative generation models for the generator, while the generation stage is not the focus of this work. Our method can not explain how the generator understands statements and concepts to generate explanations.

Acknowledgement

We thank Zuohui Fu, Dongkuan Xu and anonymous reviewers for their valuable comments.

This work was supported in part by NSF IIS-1910154, 2007907, and 2046457. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Hanxiong Chen, Yunqi Li, Shaoyun Shi, Shuchang Liu, He Zhu, and Yongfeng Zhang. 2022a. Graph collaborative reasoning. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 75–84.

Hanxiong Chen, Yunqi Li, He Zhu, and Yongfeng Zhang. 2022b. Learn basic skills and reuse: Modularized adaptive neural architecture search (manas). In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*.

Hanxiong Chen, Shaoyun Shi, Yunqi Li, and Yongfeng Zhang. 2021. Neural collaborative reasoning. In *Proceedings of the Web Conference 2021*, pages 1516–1527.

Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2019. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 765–774.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. 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)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Shijie Geng, Zuohui Fu, Yingqiang Ge, Lei Li, Gerard de Melo, and Yongfeng Zhang. 2022a. Improving personalized explanation generation through visualization.

In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 244–255.

Shijie Geng, Zuohui Fu, Juntao Tan, Yingqiang Ge, Gerard De Melo, and Yongfeng Zhang. 2022b. Path language modeling over knowledge graphsfor explainable recommendation. In *Proceedings of the ACM Web Conference 2022*, pages 946–955.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022c. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *AAAI*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, and Minlie Huang. 2020. Generating commonsense explanation by extracting bridge concepts from reasoning paths. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 248–257, Suzhou, China. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Belinda Z. Li, Maxwell Nye, and Jacob Andreas. 2021a. Implicit representations of meaning in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1813–1827, Online. Association for Computational Linguistics.

Lei Li, Li Chen, and Yongfeng Zhang. 2020a. Towards controllable explanation generation for recommender systems via neural template. In *Companion proceedings of the web conference 2020*, pages 198–202.

Lei Li, Yongfeng Zhang, and Li Chen. 2020b. Generate neural template explanations for recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 755–764.

Lei Li, Yongfeng Zhang, and Li Chen. 2021b. Extra: Explanation ranking datasets for explainable recommendation. In *Proceedings of the 44th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 2463–2469.

Lei Li, Yongfeng Zhang, and Li Chen. 2021c. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957.

Lei Li, Yongfeng Zhang, and Li Chen. 2022a. On the relationship between explanation and recommendation: Learning to rank explanations for improved performance. *ACM Transactions on Intelligent Systems and Technology*.

Lei Li, Yongfeng Zhang, and Li Chen. 2022b. Personalized prompt learning for explainable recommendation. *arXiv preprint arXiv:2202.07371*.

Bill Yuchen Lin, Minghan Shen, Wangchunshu Zhou, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *FINDINGS*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*.

Hugo Liu and Push Singh. 2004. Conceptnet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22:211–226.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ArXiv*, abs/2107.13586.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. 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 *ACL*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Kyle Richardson and Ashish Sabharwal. 2020. What does my qa model know? devising controlled probes using expert knowledge. *Transactions of the Association for Computational Linguistics*, 8:572–588.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Shaoyun Shi, Hanxiong Chen, Weizhi Ma, Jiaxin Mao, Min Zhang, and Yongfeng Zhang. 2020. Neural logic reasoning. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1365–1374.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. olmpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Juntao Tan, Shijie Geng, Zuohui Fu, Yingqiang Ge, Shuyuan Xu, Yunqi Li, and Yongfeng Zhang. 2022. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In *Proceedings of the ACM Web Conference 2022*, pages 1018–1027.

Juntao Tan, Shuyuan Xu, Yingqiang Ge, Yunqi Li, Xu Chen, and Yongfeng Zhang. 2021. Counterfactual explainable recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1784–1793.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Empirical Methods in Natural Language Processing*.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. SemEval-2020 task 4: Commonsense validation and explanation. In *Proceedings of The 14th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy. Association for Computational Linguistics.

Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 285–294.

Yikun Xian, Zuohui Fu, Handong Zhao, Yingqiang Ge, Xu Chen, Qiaoying Huang, Shijie Geng, Zhou Qin, Gerard De Melo, Shan Muthukrishnan, et al. 2020. Cafe: Coarse-to-fine neural symbolic reasoning for explainable recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1645–1654.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and E. Cambria. 2018. Recent trends in deep learning based natural language processing [review article]. *IEEE Computational Intelligence Magazine*, 13:55–75.

Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval*, 14(1):1–101.

Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 83–92.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. In *AAAI*.

Yaxin Zhu, Yikun Xian, Zuohui Fu, Gerard de Melo, and Yongfeng Zhang. 2021. Faithfully explainable recommendation via neural logic reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3083–3090.