Exploring Strategies for Generalizable Commonsense Reasoning with Pre-trained Models

Kaixin Ma[†], **Filip Ilievski**[§], **Jonathan Francis**[†¶],
Satoru Ozaki[†], **Eric Nyberg**[†], **Alessandro Oltramari**[¶]
[†]Language Technologies Institute, Carnegie Mellon University
[§]Information Sciences Institute, University of Southern California
[¶]Human-Machine Collaboration, Bosch Research Pittsburgh
{kaixinm, jmf1, sozaki, ehn}@cs.cmu.edu, ilievski@isi.edu, alessandro.oltramari@us.bosch.com

Abstract

Commonsense reasoning benchmarks have been largely solved by fine-tuning language models. The downside is that fine-tuning may cause models to overfit to task-specific data and thereby forget their knowledge gained during pre-training. Recent works only propose lightweight model updates as models may already possess useful knowledge from past experience, but a challenge remains in understanding what parts and to what extent models should be refined for a given task. In this paper, we investigate what models learn from commonsense reasoning datasets. We measure the impact of three different adaptation methods on the generalization and accuracy of models. Our experiments with two models show that fine-tuning performs best, by learning both the content and the structure of the task, but suffers from overfitting and limited generalization to novel answers. We observe that alternative adaptation methods like prefix-tuning have comparable accuracy, but generalize better to unseen answers and are more robust to adversarial splits.

1 Introduction

Machine commonsense reasoning has recently gained new traction, largely due to a collection of diverse benchmarks (Talmor et al., 2019; Bhagavatula et al., 2019; Sap et al., 2019) and the successful application of language modeling methods on these benchmarks (Ma et al., 2019; Shwartz et al., 2020; Bauer and Bansal, 2021). The most widely adopted approach to solve these commonsense reasoning tasks is by fine-tuning large pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019) on the task-specific training data. Meanwhile, it has been shown that language models are able to acquire certain commonsense background knowledge, during their pre-training on large textual data (Petroni et al., 2019; Davison et al., 2019; Ma et al., 2021). In light of these findings and the large capacity of these language models, recent work has proposed lightweight alternatives to fine-tuning LMs, e.g., by only updating a small amount of additional parameters (Lin et al., 2020b; Li and Liang, 2021), or by updating the inputs while keeping the model weights intact (Jiang et al., 2020; Shin et al., 2020). Intuitively, these lightweight methods may retain the model's pre-trained knowledge to a large extent, and elicit the suitable knowledge for the target task, provided that much of this knowledge has already been encoded in the model parameters. However, to our knowledge, no comprehensive comparison exists between these model updating strategies.

In this paper, we pose the question: *What do models learn from commonsense reasoning datasets?* We consider three representative learning methods: regular fine-tuning, model extension with prefix-tuning (Li and Liang, 2021), and model prompting with Autoprompt (Shin et al., 2020). We apply them to two representative model classes: the auto-regressive language model GPT-2 (Radford et al., 2019) and sequence-to-sequence language model BART (Lewis et al., 2020). We conduct thorough evaluation on the generative evaluation benchmarks ProtoQA (Boratko et al., 2020) and CommonGen (Lin et al., 2020a), by training on different partitions of the training data. Our experiments show that fine-tuning performs best, by learning both the content and the structure of the task, but suffers from overfitting and limited generalization to novel answers. Prompting methods have lower accuracy, but tend to show higher robustness to "adversarial" splits. Extending the models by prefix-tuning represents a "sweet spot" between task accuracy, generalization, and robustness.

2 Related Work

Prior works probe the commonsense knowledge learned by the LMs. Davison et al. (2019) mined

commonsense knowledge from LMs, using templates with masked tokens; Richardson and Sabharwal (2020) designed diagnostic tasks to probe LMs' knowledge of definitions and taxonomic reasoning. The LAMA probes (Petroni et al., 2019) demonstrate that LMs can largely recover knowledge in existing (commonsense) knowledge graphs: they could thus be queried/prompted directly as knowledge bases (Shwartz et al., 2020; Shin et al., 2020). Ettinger (2020) diagnoses the BERT model, finding that it struggles with complex inference, role-based event prediction, and grasping the contextual impacts of negation. The logical commonsense probes in RICA (Zhou et al., 2020) show that LMs perform similar to random guessing in the zero-shot setting, they are heavily impacted by statistical biases, and are not robust to linguistic perturbations. Elazar et al. (2021) posit that while LMs can learn to perform well on commonsense tasks, their commonsense reasoning ability mostly comes from fine-tuning on the task data.

Some works have sought to uncover what models learn through training on question answering datasets, exposing various dataset artifacts in the process (Jia and Liang, 2017; Kaushik and Lipton, 2018; Pugaliya et al., 2019). Welbl et al. (2020) found that models trained on the SQuAD2.0 dataset (Rajpurkar et al., 2018) are insensitive to the meaningful changes in the question and predict the same answer. Ko et al. (2020) found that BERT easily picks up the position bias in the SQuAD dataset (Rajpurkar et al., 2016) and models' performance can drop by more than 50 points on *f1*-score when training on a biased subset. Sen and Saffari (2020) analyzed the model's ability to generalize, by training on 5 different QA datasets, and found that no single dataset is robust to perturbations in the questions. Shah et al. (2020) tested models, trained on several multiple-choice QA datasets, and showed that they are largely relying on dataset biases. Previous work mostly studies the language models, as-is, or evaluated models fine-tuned on the QA datasets. In this paper, we go a step further and investigate the models adapted to a target task, using three different methods, and we study their effect on the model's learning process.

3 Experimental Setup

3.1 Task and datasets

We experiment with generative commonsense tasks, assuming that they are more realistic to real-world deployment of LMs and that they provide more insight about models' reasoning abilities. Specifically, we evaluate our models on the recently-introduced ProtoQA (Boratko et al., 2020) and CommonGen (Lin et al., 2020a) datasets. For ProtoQA, given a question about a prototypical situation, the model is expected to produce a ranked list of answers. Each question in the dev and test sets is annotated with 100 answers, which are further manually grouped into clusters: the model's outputs are compared with the answer clusters, and the scores reflect the sizes of the matched clusters. We adopt ProtoQA's official evaluation metrics: `Max answer@`k (percentage of correct answers with top-k predictions) and `Max Incorrect@`k (percentage of correct answers after making k mistakes). We compute the answer matches based on WordNet similarity, as recommended by Boratko et al. (2020). For CommonGen, given a set of 3-5 input concepts, the task is to generate a scene description, utilizing all input concepts. Following prior work, we adopt `BLEU` (Papineni et al., 2002), `ROUGE` (Lin, 2004), `METEOR` (Banerjee and Lavie, 2005), `CIDEr` (Vedantam et al., 2015) and `SPICE` (Anderson et al., 2016), as evaluation metrics.

3.2 Strategies

We describe how we adapt pre-trained GPT-2 and BART models to a target task with three methods[1]:

(S1) Fine-tuning is the classic model adaptation approach, where all its parameters are updated using the training signal from the ground truth.

(S2) Prefix-tuning (Li and Liang, 2021) is a method which fixes the pre-trained model's parameters during adaptation. This method adds trainable parameters, called prefix states, to the self-attention component (Vaswani et al., 2017) of every transformer layer in the model; only these prefix states are updated during training. Essentially, the prefix states act as conditioning variables that contextualize the representation of the inputs, such that the model can generate the desired outputs.

(S3) Instead of updating model parameters, *Autoprompt* (Shin et al., 2020) appends a few trigger-tokens to the input and updates these trigger-tokens during training. Specifically, the gradient with respect to the trigger-tokens is computed using the ground-truth data. During training, new trigger-tokens are discovered, along the direction of the

[1]Our code is available at https://github.com/Mayer123/CS_Model_Adaptation

Table 1: Results on the ProtoQA dev set, we ran experiments 5 times with different seeds, and report 95% confidence interval. *Note that the human performance accuracies from (Boratko et al., 2020) are reported on the test set, we assume that the accuracy values would be similar on dev set.

Model	Ans@1	Ans@3	Ans@5	Ans@10	InCorr@1	InCorr@3	InCorr@5
GPT2	28.2	27.1	27.2	30.7	14.4	21.1	27.5
GPT2-Autoprompt	25.5(±2.5)	30.9(±2.8)	35.1(±1.7)	42.4(±1.7)	14.7(±2.6)	28.7(±1.9)	35.5(±1.3)
GPT2-Prefix-tuning	42.7(±2.0)	**51.5(±1.4)**	52.2(±0.8)	60.8(±0.5)	28.8(±1.3)	47.6(±0..9)	56.9(±0.8)
GPT2-Finetune	**49.3(±1.7)**	50.3(±1.5)	**53.0(±2.2)**	**63.0(±0.6)**	**31.9(±1.4)**	**49.9(±1.4)**	**57.9(±1.4)**
BART	20.9	29.8	32.2	37.5	15.1	27.3	32.2
BART-Autoprompt	28.2(±4.7)	33.8(±0.9)	37.2(±1.5)	44.6(±3.0)	16.6(±2.1)	31.1(±2.3)	38.9(±2.7)
BART-Prefix-tuning	45.5(±2.9)	51.0(±1.9)	54.8(±1.6)	**62.9(±1.0)**	32.7(±1.4)	51.4(±0.9)	58.7(±1.0)
BART-Finetune	**53.6(±2.5)**	**54.3(±2.2)**	**56.3(±0.9)**	62.6(±1.0)	**35.6(±0.6)**	**53.9(±1.5)**	**59.5(±1.8)**
Human*	78.4	74.4	72.5	73.3	55.8	69.4	72.4
Single Human*	40.5	39.4	41.0	45.6	23.9	36.0	40.5

gradient, to replace the existing ones and to minimize the loss. Essentially, this method automatically learns to paraphrase the input question so that the model can generate the desired outputs.

We select fine-tuning, prefix-tuning, and Autoprompt as they are representative methods for adapting a pre-trained model to a target task, namely: 1. model adaptation (fine-tuning); 2. model extension (prefix-tuning); and 3. input adaptation (Autoprompt). We illustrate whether the model has learned different behaviors from methods with different degrees of adaptation. Training details can be found in the appendix A.1.

3.3 Research Questions

We address three questions in this paper, namely: *(RQ1: Adaptation level) How do different levels of adaptation affect the model's task-specific performance?* We expect that methods that adapt a larger number of parameters to the training task (fine-tuning) would perform better on the task itself, as the larger search space makes it more likely to find a task optimum. We investigate this by comparing S1-S3 on the two benchmarks.

(RQ2: Task structure) Do models only learn the task structure during training? As we are working with relatively small benchmarks, we hypothesize that LMs acquire most of the necessary commonsense knowledge during pre-training instead of at adaptation time, during which they instead learn to *elicit* this knowledge. In this case, such adaptation to task structure could be done on just a subset of the training data without a large drop in performance, and the model need not depend on any lexical similarities between the training set and the dev set. To this end, we train our models with each adaptation method on: 1) a *non-overlap* subset of ProtoQA, consisting of train-set QA pairs whose answers do not have any vocabulary overlap with the dev set answers; and 2) a *min-overlap* split for CommonGen, selecting training instances whose input concepts appear at most once in the dev set.

Table 2: Summary of the dataset splits.

Dataset	Subset	# of examples
ProtoQA	Full-data	44,964
ProtoQA	*non-overlap*	18,855
ProtoQA	*similarity*	17,461
CommonGen	Full-data	67,398
CommonGen	*min-overlap*	17,914
CommonGen	*random*	17,914

(RQ3: Novelty) Do models simply memorize the training data, or do they learn to reason on novel questions and answers as well? To test whether models merely retrieve lexically similar examples, we formulate a *similarity* subset on ProtoQA, comprised of the 100 questions in the training set with the highest cosine similarity for every dev set question. To test whether models achieve better performance due to improved *reasoning* ability, we selected 30 questions from the ProtoQA dev set, where the model answers are at least partially correct, and we minimally changed the question through manual annotation—so that the required reasoning process is the same, but the answer set is different (example question pairs are given in the appendix A.2). Then, we use the BART model trained with each adaptation method to generate answers for the 30 new questions. We manually validate the 30 new questions and the 30 original questions, in order to compute the accuracy of the models as well as the percentage of overlapping answers between the original questions and new questions. More details are provided in section

4.1.1, and a summary of all the dataset splits used in our experiments is shown in Table 2.

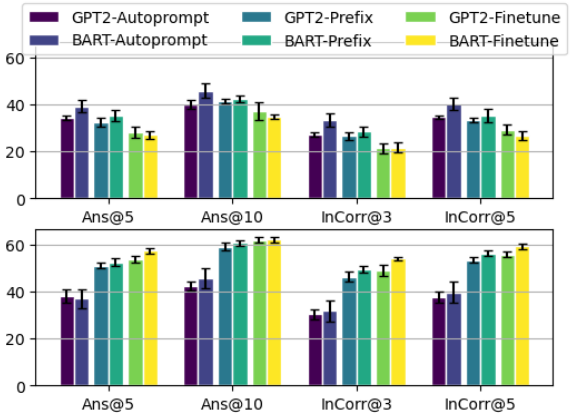

Figure 1: Results on ProtoQA dev set. The upper figure shows the models trained on the *non-overlap* subset. The bottom figure shows the model trained on the *similarity* subset.

4 Results

4.1 ProtoQA

In response to *RQ1 (adaptation level)*, Table 1 shows that prefix-tuning yields similar or slightly worse results compared to fine-tuning, for both LM classes, indicating that prefix-tuning is a promising lightweight alternative to fine-tuning. Autoprompt lags behind the tuning methods, while outperforming the zero-shot baseline. This is not surprising, as Autoprompt performs a fairly limited adaptation by only updating trigger tokens in a discrete space.

The results for *RQ2 (task structure)* are shown in Figure 1. Fine-tuning a model on the *non-overlap* data leads to a drastic drop in performance, compared to using the full training data. Prefix-tuning's drop in performance is smaller than that of fine-tuning, while Autoprompt achieves the best performance when training on this subset. The result of Autoprompt is similar to training on the full data, showing that Autoprompt is much more robust towards an adversarial training split and is mainly learning how to elicit the model's pre-trained knowledge to answer the questions. Fine-tuning is learning knowledge together with the task structure, while prefix-tuning stands between fine-tuning and Autoprompt. Since prefix-tuning does not change the pre-trained model's parameters, but rather adds new parameters, it learns to mix the knowledge gained from pre-training with the signal from training instances to answer new questions.

For *RQ3 (novelty)*, the results of training on the *similarity* subset are shown in Figure 1. Although the number of QA pairs is much lower, fine-tuning achieves the same results as in the full-data setting. This shows that fine-tuning benefits more from the content of the training data than the task format, further informing our findings for *RQ2*. Prefix-tuning performs slightly worse than the full data setting, indicating that here it is largely learning the training content. Autoprompt achieves similar results as in the full-data and *non-overlap* settings, confirming our *RQ2* observations that models are only learning the task format. We note that, while retrieving lexically similar questions might yield partial results, this form of pattern-matching is insufficient for commonsense reasoning. For example, for a training question *Name a vegetable that people like to steam*, the model learned the answer *cauliflower*, which is coincidentally also a correct answer to the dev question *Name a vegetable that is as large as your head*. In other words, the model answers correctly for the wrong reasons.

Table 3: Human evaluation results

Measurement	Fine-tune	Prefix	Autoprompt
Original question	54.8	53.6	43.7
New question	42.3	42.3	30.9
% overlap answers	44.7	44.7	38.1

4.1.1 Manual Annotation

We conduct manual annotation, to further verify our observations for *RQ3*. For the BART model, trained with each adaptation method, we generate top10 answers for every question and then annotate each answer, independently. We annotate each answer on a 5-point Likert scale, where 1 means strongly disagree, 2 means mostly disagree, 3 means not sure/it depends, 4 means mostly agree, and 5 means strongly agree. In total, 4 researchers annotated 1165 QA pairs where each QA pair received 3 ratings. The overall Kripendorf alpha (Krippendorff, 2004) score is 0.52, which is moderate agreement. If we merge answers choices 1 and 2 to be 'incorrect' and merge 4 and 5 to be 'correct', and then compute the 3-class categorical agreement score using Fleiss kappa, the agreement score is 0.36, which is fair agreement. Then we consolidate the 3 ratings by taking the average of the 3 annotations and consider an answer to be correct if the average score is greater than 3.5.

Table 4: Results on CommonGen dev set with the BART model.

Method	Split	Bleu4	METEOR	ROUGE-L	CIDEr	SPICE
Autoprompt	Full-data	0.3(±0.2)	8.9(±3.0)	16.2(±4.1)	1.0(±0.5)	-
Prefix	Full-data	34.4(±0.4)	**33.0**(±**0.2**)	53.9(±0.2)	**17.4**(±**0.2**)	**33.4**(±**0.3**)
Fine-tune	Full-data	**34.9**(±**0.4**)	32.9(±0.1)	**54.1**(±**0.1**)	**17.4**(±**0.1**)	33.0(±0.2)
Autoprompt	*Min-overlap*	0.3(±0.2)	8.0(±2.1)	16.5(±2.3)	1.1(±0.6)	-
Prefix	*Min-overlap*	**32.8**(±**1.0**)	**31.5**(±**0.3**)	**52.1**(±**0.5**)	**16.6**(±**0.3**)	**32.2**(±**0.2**)
Fine-tune	*Min-overlap*	31.5(±0.2)	30.6(±0.3)	51.2(±0.2)	15.9(±0.1)	30.7(±0.5)
Autoprompt	*Random*	0.1(±0.1)	7.4(±3.1)	14.5(±4.2)	1.1(±0.8)	-
Prefix	*Random*	33.4(±0.3)	**32.3**(±**0.4**)	52.9(±0.3)	**17.2**(±**0.2**)	**32.8**(±**0.7**)
Fine-tune	*Random*	**34.1**(±**0.3**)	32.1(±0.1)	**53.3**(±**0.2**)	17.1(±0.2)	32.3(±0.4)

The results from manual assessment of the models' reasoning capabilities are shown in Table 3. We observe that our LMs are not able to capture subtle changes in the question that lead to a different answer set; models are getting worse performance on the new questions, overall. We believe this is because the newly-generated questions are more difficult to answer, as they seldom appear in any text corpus in general. We also see a high overlap between the generated answers to the original and the newly-created questions, especially for fine-tuning and prefix-tuning, where nearly half (44.7%) of the answers are repeated. This confirms our observation that models memorize/retrieve training-set answers without actually engaging in reasoning.

4.2 CommonGen

The full results on the CommonGen dataset are shown in Table 4. Overall, we can see that the results follow a similar trend to those of ProtoQA, as prefix-tuning is able to perform significantly better than fine-tuning when trained on an 'adversarial' split. We notice that the relative drop of performance for both methods on the *Min-overlap* subset is less drastic than that of ProtoQA. We think this is mainly due to the task format. For ProtoQA, models need to perform one or a few hops of reasoning to answer the questions and there is no direct evidence from the question itself, i.e., the model cannot directly copy answers from the questions. However, the model is directly given the target concepts as inputs for CommonGen, which the model can directly use as its outputs. Thus, we argue that the amount of reasoning required in CommonGen is more restricted than in ProtoQA, and models are less likely to leverage the clues to solve the task.
Also, it is worth noting that the accuracy of Autoprompt is extremely low on all 3 splits. In fact, Autopropmt fails to generate any meaningful sentences after training, and the `SPICE` metric could not be computed. We, again, attribute this to the task format. Autoprompt would eventually discover tokens that are meaningless to humans, and we can think of them as injecting task-specific noise to the pre-trained models. For ProtoQA, the model is expected to generate single words or short-phrase answers to complete the sentence, i.e., converted question, thus it is reasonable for the model to do it even with the injected noise. However, for CommonGen, the model is expected to generate a full sentence as output; with Autoprompt, the task basically translates to generating a sentence given input concepts and a bunch of random tokens, which is very different from BART's pre-training context.

5 Conclusions

Experiments with two language model classes, on two generative commonsense benchmarks, under three adaptation methods, revealed that the learning efficiency of LMs relies heavily on the adaptation method. Fine-tuning teaches the model both the structure of the task and the content, prompting approaches focus on learning the task structure only, while model extension by prefix-tuning falls between these two extremes. Consequently, prompting is the least sensitive of the three methods to the training data size and quality, and prefix-tuning can generalize better to novel concepts regardless of the task format. Future work on generalizable common sense leverage these findings, and: 1) avoid fine-tuning, as we may never be able to create datasets without any unintended biases (Linzen, 2020); and 2) evaluate on multiple independent test-sets to better replicate real-world settings, as training on any split of data can lead to an overestimation of performance (Søgaard et al., 2021).

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: semantic propositional image caption evaluation. *CoRR*, abs/1607.08822.

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.

Lisa Bauer and Mohit Bansal. 2021. Identify, align, and integrate: Matching knowledge graphs to commonsense reasoning tasks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2259–2272, Online. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. 2019. Abductive commonsense reasoning. *arXiv preprint arXiv:1908.05739*.

Michael Boratko, Xiang Li, Tim O'Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. 2020. ProtoQA: A question answering dataset for prototypical commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1122–1136, Online. Association for Computational Linguistics.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*, pages 4171–4186.

Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. 2021. Back to square one: Bias detection, training and commonsense disentanglement in the winograd schema. *arXiv preprint arXiv:2104.08161*.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

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.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 2020. Look at the first sentence: Position bias in question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1109–1121, Online. Association for Computational Linguistics.

Klaus Krippendorff. 2004. *Content Analysis: An Introduction to Its Methodology (second edition)*. Sage Publications.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020a. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020b. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 441–459, Online. Association for Computational Linguistics.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, Online. Association for Computational Linguistics.

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.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proc. of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari. 2021. Knowledge-driven data construction for zero-shot evaluation in commonsense question answering. *In Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21)*.

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*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

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*.

Hemant Pugaliya, James Route, Kaixin Ma, Yixuan Geng, and Eric Nyberg. 2019. Bend but don't break? multi-challenge stress test for QA models. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 125–136, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proc. of EMNLP-IJCNLP*, pages 4463–4473.

Priyanka Sen and Amir Saffari. 2020. What do models learn from question answering datasets? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2429–2438, Online. Association for Computational Linguistics.

Krunal Shah, Nitish Gupta, and Dan Roth. 2020. What do we expect from multiple-choice QA systems? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3547–3553, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4615–4629, Online. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proc. of NAACL*, pages 4149–4158.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *CVPR*, pages 4566–4575. IEEE Computer Society.

Johannes Welbl, Pasquale Minervini, Max Bartolo, Pontus Stenetorp, and Sebastian Riedel. 2020. Undersensitivity in neural reading comprehension. *CoRR*, abs/2003.04808.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Pei Zhou, Rahul Khanna, Seyeon Lee, Bill Yuchen Lin, Daniel Ho, Jay Pujara, and Xiang Ren. 2020. Rica:

Evaluating robust inference capabilities based on commonsense axioms. *arXiv preprint arXiv:2005.00782*.

A Appendix

A.1 Training details

A.1.1 Hyper-parameters

For all of our experiments on ProtoQA, we follow the baseline model setup of Boratko et al. (2020) and train the model with learning rate 1e-5, batch size 8, warm-up steps 150 and Adam epsilon 1e-6, unless otherwise specified. We trained the model for the amount of steps that are equivalent to 1 epoch of training on full training set, i.e., we train the model for longer epochs on *non-overlap* and *similarity* subsets. For both training and inference, we adopted the same question-converting templates as Boratko et al. (2020). During inference, we do nucleus sampling with top-p=0.9, temperature 0.69 to generate 300 answer candidates; we group them and rank them based on frequency.

For all of the experiments on CommonGen, we used learning rate of 1e-5, batch size 16, warm-up steps 500 and Adam epsilon 1e-6. We trained the model for 2 epochs and similarly we train models for longer epochs on *min-overlap* and *random* subsets. During inference, we do beam search with beam size 5, length penalty 0.6, and repetition penalty 2.0. Note that we disabled positional embeddings in the BART encoder, for all CommonGen experiments, as we found them detrimental to model performance.

A.1.2 Model Implementation

We used the BART-large and GPT2-large model provided by the transformers library (Wolf et al., 2019). For prefix-tuning, we used prefix with length 10 and a 1 layer of prefix MLP with hidden size 512 (we tried {512, 800} and found them to have very close results). The learning rate is 5e-5, while other hyperparameters are the same as in fine-tuning (we tried {1e-5, 2e-5, 5-e5, 8e-5}, and found the latter 2 achieve slightly better results). For prefix-tuning with the BART model, we added prefix states to self-attention in encoder layers and self-attention and cross-attention in decoder layers. For GPT2 model, we only add prefix states to self-attention in decoder layers.

For Auto-prompt with BART, we used the same 10 trigger tokens for both encoder and decoder; the trigger tokens are all initialized with mask tokens. For GPT2, we also used 10 trigger tokens. Since the model does not have mask tokens, we initialized triggers with the tokenized prompt "Based on simple commonsense fact, we know that", which is exactly 10 tokens by BPE. We train the models with batch size 32 and gradient accumulation steps 4 (we tried batch sizes {32, 128, 256} and found that larger batch size yield more stable results). At each update step, we search for the next trigger token, within the 100 closest candidate tokens, along the gradient direction (we used 10 candidate tokens for CommonGen experiments, as we found that both 10 and 100 lead to extremely bad results—so we used 10 to save computation time).

A summary of the number of trainable parameters for each model-adaptation method combination is shown in Table 5.

Table 5: Trainable parameters comparison

LM	Adaptation	# Parameters
GPT2	Fine-tune	774M
GPT2	Prefix-tuning	900K
GPT2	Autoprompt	0
BART	Fine-tune	400M
BART	Prefix-tuning	1.44M
BART	Autoprompt	0

A.1.3 Dataset splits

The ProtoQA dataset provides a dev-scraped set and a dev-crowdsourced set, where dev-scraped is collected from the Family-feud fan website, i.e., same as the training set, while the dev-crowdsourced set contains newly written questions and answers by crowd-workers, i.e., same as the test set. We select the best model using the loss on dev-scraped set and report results on the dev-crowdsourced set, because the test set answers are hidden and we need the ground-truth answers to test our hypothesis. In the main paper content, all references of ProtoQA dev-set refer to the dev-crowdsourced set. For CommonGen dataset, we select best models, using the loss on the dev-set.

For the *similarity* subset of ProtoQA, we adopt the `stsb-roberta-large` model from the sentence-transformer (Reimers and Gurevych, 2019) library and compute the cosine similarity between the train and the dev questions.

A.2 New Questions

Examples of the original and newly written questions, along with the model predictions are shown in Table 6.

Table 6: Example of original questions and newly written questions and the corresponding predictions from BART model trained on Full data, the bold answers are corrected ones selected by human evaluation

Type	Question/Answers
Original	**Name something a monk probably would not own.**
Fine-tune	**car**, clothes, money, bike, **computer**, horse, **cell phone**, shoes, **motorcycle**, bicycle
Prefix-tuning	**car**, **motorcycle**, bike, clothes, money, robe, shoes, **sword**, horse, **camera**
Autoprompt	money, **car**, **gun**, **sword**, **boat**, wine, shoes, coffee, **beer**, cat
New	**Name something a monk probably would own.**
Fine-tune	**pen**, cross, **book**, sword, **rosary**, brooms, bible, **robe**, **books**, kite
Prefix-tuning	**robe**, bible, cross, **pen**, **beard**, rosa, sword, **robes**, **monastery**, apron
Autoprompt	money, bike, sword, **chair**, **books**, **brooch**, bicycle, gold, phone, kitty
Original	**Name a vegetable that is about as big as your head..**
Fine-tune	broccoli, carrot, **cabbage**, **cauliflower**, carrots, **lettuce**, spinach, **cucumber**, potato, corn
Prefix-tuning	**cucumber**, broccoli, **cabbage**, carrot, **lettuce**, **pumpkin**, spinach, beet, **cauliflower**, celery
Autoprompt	**cabbage**, broccoli, spinach, **lettuce**, carrots, **cauliflower**, potato, potatoes, **pumpkin**, gooseberries
New	**Name a vegetable that is about as big as your fist.**
Fine-tune	broccoli, cabbage, carrot, carrots, spinach, lettuce, cauliflower, cucumber, corn, **beet**
Prefix-tuning	cucumber, broccoli, cabbage, lettuce, spinach, **beet**, celery, pumpkin, carrot, cauliflower
Autoprompt	broccoli, cabbage, lettuce, carrots, spinach, squash, cucumbers, eggplant, cucumber, **kiwi**
Original	**Name a sport that requires a lot of equipment.**
Fine-tune	**hockey**, **golf**, **football**, **tennis**, basketball, **baseball**, soccer, boxing, wrestling, volleyball
Prefix-tuning	**football**, basketball, **hockey**, soccer, **tennis**, **golf**, **baseball**, wrestling, volleyball, **skiing**
Autoprompt	**hockey**, soccer, **golf**, basketball, **football**, **tennis**, baseball, **rugby**, volleyball, **ice hockey**
New	**Name a sport that you don't need a lot of equipment for.**
Fine-tune	hockey, tennis, baseball, golf, **soccer**, football, **basketball**, **bowling**, **volleyball**, **swimming**
Prefix-tuning	**basketball**, tennis, **soccer**, golf, football, hockey, baseball, **bowling**, **swimming**, skiing
Autoprompt	**basketball**, **soccer**, hockey, golf, **volleyball**, tennis, football, baseball, rugby, lacrosse
Original	**Name something around the house that's often replaced.**
Fine-tune	tv, television, furniture, **dishes**, carpet, **toilet paper**, refrigerator, windows, stereo, **lights**
Prefix-tuning	carpet, lamp, light, furniture, tv, clothes, **dishes**, television, **bedding**, **toilet paper**
Autoprompt	TV, tv, couch, table, toilet, bed, television, microwave, chair, lamp
New	**Name something around the house that's hardly ever replaced.**
Fine-tune	tv, television, **furniture**, dishes, **refrigerator**, stereo, **carpet**, toilet paper, **windows**, **appliances**
Prefix-tuning	dishes, **furniture**, lamp, **carpet**, tv, clothes, bedding, TV, light, television
Autoprompt	TV, tv, **fridge**, **microwave**, **couch**, **refrigerator**, **dishwasher**, coffee table, **bed**, **table**
Original	**Name a job where you have to be awake at night.**
Fine-tune	**police officer**, doctor, **nurse**, **security guard**, lawyer, teacher, **firefighter**, construction, actor, **cop**
Prefix-tuning	**police officer**, **nurse**, construction, doctor, **security guard**, **bartender**, waiter, babysitter, **firefighter**, teacher
Autoprompt	construction, work, carpenter, **firefighter**, **truck driver**, roofing, **police**, **fireman**, **bartender**, school
New	**Name a job where you only have to work during the day.**
Fine-tune	nurse, **teacher**, police officer, **doctor**, bartender, construction, waitress, **lawyer**, waiter, **mechanic**
Prefix-tuning	nurse, **teacher**, **lawyer**, **doctor**, bartender, waiter, **mechanic**, construction, **sales**, waitress
Autoprompt	construction, hospital, fireman, restaurant, **plumber**, cook, **cleaning**, chef, firefighter, **teaching**