Chain-of-Thought finetuning for smaller language models without using a teacher model

Kieron Kretschmar, Elias Dubbeldam, Jonathan Gerbscheid, Oresitis Gorgogiannis, Robin Sasse
{kieron.krestschmar, e.f.dubbeldam, jonathan.gerbscheid, orestis.gorgogiannis, robin.sasse}@student.uva.nl

Abstract

Large Language Models are typically used in a zero-shot or few-shot setting, that is, on novel tasks. When prompted to only give an answer, these models usually do not perform to their full potential. Chain-of-Thought prompting has been shown to successfully extract reasoning capabilities for large models (>100B parameters). However, smaller models do not inherently show these capabilities, even when prompted in such a manner. We examine whether these models can be finetuned in a few-shot Chain-of-Thought setting.

Our method only requires a small number of ground truth Chain-of-Thought examples. Unlike other methods, we train on question-answer pairs only, while reusing the same static support for each forward pass. We show that Chain-of-Thought finetuning in this manner can improve the answers to two common reasoning tasks (3-digit-division and natural language inference) for models as small as 560m parameters. However, our technique does not outperform finetuning without CoT, indicating that the models do not learn to reason. We attribute most performance gains to finetuning on the task itself.

1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020a), OPT (Zhang et al., 2022), BLOOM (Scao et al., 2022), T5 (Raffel et al., 2020), or LLaMa (Touvron et al., 2023) are typically used for few-shot or zero-shot tasks such as question answering, common sense reasoning, or even simple arithmetic (Suzgun et al., 2022). However, when only prompted to give an answer, these models usually underperform when faced with challenging reasoning tasks, such as arithmetic reasoning or natural language inference (NLI) tasks.

To improve an LLM's performance in these tasks, previous research has been investigating the use of Chain-of-Thought (CoT) prompting. The simplest, successful approach to extract the model's zero-shot CoT reasoning capabilities is to simply append "Let's think step by step" to an instruction (Kojima et al., 2022).

While this method already improves the model's accuracy by a big margin on a wide range of linguistic tasks, other researchers have shown that LLMs can benefit even more from the one-shot or few-shot setting (Brown et al., 2020b). In this setting, the input to the model consists of several exemplary question-answer pairs for the task plus a novel question. For the latter, the model has to provide an answer. However, this method has shown to be less effective for tasks requiring reasoning capabilities.

A more effective way of extracting the model's few-shot capabilities for these tasks is to use CoT prompting in the few-shot setting (Wei et al., 2022; Suzgun et al., 2022). In this setting the model's input consists of several exemplary question-answer-explanation pairs for the task plus a novel question for which the model has to provide an answer. However, CoT reasoning capabilities only emerge empirically for very large models at around 100B parameters.

Instruction tuning can unlock this capability for relatively small models at around 1B parameters or below. Magister et al. (2022) show that finetuning these smaller language models with CoT, generated by a larger teacher model, can improve the performance of smaller language models across a wide range of downstream reasoning tasks. In this work, we explore the possibility of using a smaller set of high-quality humanly annotated explanations to build static CoT prompts for training. We ask the following research questions:

- Can we uncover CoT reasoning capabilities for smaller language models by finetuning with a small static subset of CoT examples in a few-shot setting?

- If this approach works, at which model size does this behavior begin to appear?

2 Related work

Wei et al. (2022) show that (arithmetic) reasoning tasks can benefit from CoT in the few-shot setting. Nonetheless, they also find that smaller models, i.e. models that are much smaller than 100B parameters, do not benefit from CoT prompting. They argue that CoT reasoning is an emergent ability to increase model scale. For large models, CoT prompting does not require finetuning but rather works with in-context learning techniques. They further argue that CoT prompting enables generalization to unseen tasks.

Recently, researchers have explored the possibility of finetuning smaller language models, i.e. models of size over 1B parameters, on question-explanation-answer pairs. Magister et al. (2022) used teacher forcing on a student model. Their student models were trained on targets consisting of CoT and answer. In this setting, the CoTs were provided by a larger teacher model, in this case, GPT-3 175B.

A similar approach was recently pursued by Kim et al. (2023), who created a novel dataset for CoT finetuning. In their work, they also show that finetuning a model on their dataset can lead to improvements in zero-shot and few-shot capabilities of smaller language models for novel tasks. Since this dataset came out at the end of our project, we have not had sufficient time to incorporate it into our experiments. Nonetheless, it seems to be a promising candidate for future research in this direction.

Current approaches focus on finetuning on CoT and answer (Magister et al., 2022; Ho et al., 2022; Kim et al., 2023). What separates these works from ours is that all of them pursue finetuning the model on the CoT targets. We want to use the CoT annotations as a few-shot support to the input for the model, similar to in-context learning. We finetune on the answers only. Hence, a wrong explanation that leads to a correct answer is also allowed. To our knowledge, there are no other works that use CoTs as support while finetuning on a larger corpus of question-answer pairs.

3 Methods

3.1 Models

For this study, we choose to train different sizes of Bloom (Scao et al., 2022), as the model is publicly available and comes in various sizes between 560m to 176B parameters. Due to limited computational power, we will limit our research to the models of sizes 560m, 1B + 100m, 1B + 700m, and 3B parameters. However, CoT finetuning empirically works with the smaller models as well (Magister et al., 2022; Kim et al., 2023).

We chose Bloom over other models, such as T5 (Raffel et al., 2020) since it produced interpretable results in the few-shot setting. Other models did not provide answers that were interpretable without non-trivial manipulations of the output. Since any non-trivial design decision would interfere with the reliability of the results, we decide to go with the model that reduced the number of required decisions the most.

Another advantage of, but not limited to Bloom is, that the model has not been previously instruction trained. Any instruction pretraining would interfere with the effect of CoT finetuning. Therefore, we chose this requirement for the model.

3.2 Finetuning LoRA

Because of the size of current language models and the resulting infeasibility of fully finetuning the whole model, various lightweight finetuning strategies have been proposed (e.g. Adapters by Houlsby et al. (2019) or Prompt-Prefix-Tuning by Li and Liang (2021)). In this project, we use LoRA (Hu et al., 2021), which has recently gained popularity in the NLP community due to its simplicity and effectiveness.

When finetuning with LoRA, we usually observe that we need to update less than 1% of the parameters. In many cases, this number is even down to 0.1% for the language models in our setting and can get as low as 0.01% for LLMs such as GPT-3 (Hu et al., 2021).

Another desirable property of LoRA is that with an increase in the number of parameters, the finetuning procedure roughly converges to training the entire model. This property is not shared by other methods, such as adapters. Furthermore, no additional inference latency is introduced as we store $W = W_0 + BA$ with BA describing the bottleneck linear layer which is LoRA, and W_0 describ-

ing the original weights. Lastly, compared to finetuning prefixes, LoRA is easier to optimize.

3.3 Finetuning without ground truth Explanations

One difficulty in the CoT few-shot setting stems from the circumstance that the model is expected to generate an explanation before the answer, but no ground truth explanation is available to finetune the model on. In this section we will elaborate on our approach, the difficulties we encountered, and how we resolved them.

3.3.1 Notation

In this section, we will use the following notation: Q, E, and A denote the strings that introduce a question, explanation, and answer, respectively. We use Q ="Q:", E="Explanation:" and A ="A:". q and a denote a question and its correct answer, according to the training dataset. Explanations, i.e. the CoTs, are referred to as e. S denotes the support, a string consisting of multiple question-explanation-answer examples with explanations taken from Lampinen et al. (2022). For example in the 2-shot setting, S would have the form $S = QqEeAa\ QqEeAa$. Note that S remains fixed throughout our experiments.

3.3.2 Our approach

For each training step with a training sample (q, a), we first translate our problem into the classical setting of finetuning a language model. For this we prompt the model with $SQqE$ and let it generate an output O of 50 tokens, using the greedy generation strategy. Ideally, O has the form eAa, i.e. an explanation e followed by the answer marker A and the answer a. We attempt to extract the model's explanation e, by taking everything in O that comes before the answer marker A.

Having obtained e, we create the prompt $P = SQqEeA$ and finetune the model towards generating the correct answer a.

Note that the output O does not always contain the marker A, making it impossible to extract the explanation e. In the following sections, we present typical situations in which this occurs, and our attempts to avoid them. In cases where we can not extract an explanation e, we invalidate the sample and discard it from the loss computation.

3.3.3 Models get stuck repeating themselves

Smaller language models in particular tend to get stuck in repetitive patterns. If this happens during the generation of the explanation, no A marker is generated.

This issue was most prevalent in the smallest models, but not exclusively so, as this output generated by the 3b model shows:

> [...]Explanation:201 / 67 = (201 / 67) + (67 / 67) = (3 / 67) + (4 / 67) = (3 / 67) + (4 / 67) = (3 / 67) + (4 / 67

To prevent invalidation of these samples, we have tried replacing the greedy generation strategy with beam search and a repetition penalty according to (Keskar et al., 2019). This solved the issue but made the first step much more expensive, which is why we eventually decided to disable this change for our experiments.

3.3.4 Models generate EOS-tokens before answering

One issue, occurring mostly in bigger models, is that an EOS-token is generated before the model has either generated an answer marker A or reached the maximum 50 tokens allowed token.

One example, generated by the 3b model:

> [...]Explanation:864 / 864 = 1, so 864 / 864 = 1.EOS

Our solution is to replace the EOS-token with the answer marker A, effectively treating everything that came before it as the explanation. From qualitative evaluation, this works reasonably well, so we have keep this change for the experiments.

3.3.5 Models jump straight to the answer

After finetuning for some epochs, we observe two trends occurring in models of different sizes:

1. The answer marker A is sometimes not being generated, but the answer a itself is.
2. Explanations tend to get shorter and shorter, up until the point where no explanation is generated altogether.

We attribute this to the fact that so far we only computed the loss for the answer a itself, thereby disregarding that the explanation and answer markers are essential, too.

To combat the first issue, we adapt the finetuning step to treat the A as part of the ground truth, and computing loss for its generation. Additionally, in cases where the answer is correct, we do the same for the explanation e, following the intuition that

if the answer is correct, the explanation is helpful, too. In those cases, the label effectively becomes eAa.

We observe that these changes prevent the aforementioned issues, so we keep them for the experiments.

4 Experiments

4.1 Datasets

We investigate whether we can use a small number of high-quality humanly annotated explanations to question-answer pairs of various tasks from the Big-Bench dataset[1] to finetune differently sized models on CoT reasoning. More specifically, we prepare support consisting of a few examples of question-explanation-answer triplets before asking the model a question on which the provided answer will be evaluated. This support is constant and consists of several handpicked example CoTs that are not changed during training, nor are they changed at test time. The example CoTs are taken from the dataset provided by Lampinen et al. (2022). This dataset contains 40 different tasks with 15 handpicked and humanly annotated examples from the Big-Bench dataset.

However, we still train the model on a full task of Big-Bench containing at least 60, but up to multiple tens of thousands of examples. Only the CoT support comes from the annotated subset. For calculating the loss, we only look at the answer the model produces, as Big-Bench does not contain explanations that we can evaluate. Furthermore, we do not necessarily care for the correctness of an explanation, as long as it leads to a correct answer of the model.

In this research project, we focus on two tasks from the Big-Bench dataset, namely *3-digit division* and *Presuppositions as NLI*. We specifically chose these datasets because Lampinen et al. (2022) have not just provided explanations but have also finetuned the best-performing explanations by hand. Furthermore, both of these tasks have sufficiently many examples in Big-Bench to provide a full training set. Lastly, both tasks can be considered reasoning tasks as they require a logical CoT to be solved effectively. Therefore, we should see a difference between the CoT setting and the classical few-shot setting.

Unlike Lampinen et al. (2022), who structure their dataset question-answer-explanation, we restructure their dataset to question-explanation-answer. This approach seems more promising as the model is able to use the explanation as a CoT leading to an answer, rather than a post-hoc justification for that answer. Wei et al. (2022) observe better performance when the model is prompted to provide an explanation before the answer.

[1] https://github.com/google/BIG-bench/

4.2 Ablation study

We compare the following model-input configurations with one another:

(i) The pretrained out-of-the-box model in the classical few-shot (or in-context learning) setting. In this setting, the prompt consists of n question-answer pairs plus the question we seek an answer to. Hence, the model is prompted to give an answer only but has a few examples in the input to improve its answer. That is, $P = SQqA$ where $S = QqAa\ QqAa$ in the 2-shot setting.

(ii) The pretrained out-of-the-box model in the CoT few-shot (or in-context learning) setting. That is, the input consists of n question-explanation-answer triplets. Hence, the model is prompted to give an explanation followed by an answer. That is, $P = SQqE$ where $S = QqEeAa\ QqEeAa$ in the 2-shot setting. At test time the input consists of the same CoT few-shot support seen during training. Effectively, the model is prompted to give an explanation followed by an answer.

To set a baseline, we vary n from 0 to 5 for experiment (i) as described above. This gives us insight how the model performs without any CoT support or finetuning. The results are discussed in Section 5.1.

To study how finetuning on CoTs increases performance, we finetune the model with LoRA without CoTs (experiment (i)) and with CoTs (experiment (ii)) in $n = 4$-shot setting. If the CoT-finetuned model outperforms this model, we can be more decisive in our analysis of the effects of CoT finetuning. The results are discussed in Section 5.2.

In any setting, we only evaluate the performance on the given answer. If the model is prompted to give an explanation, this CoT is ignored when computing for accuracy. A wrongful explanation is never penalized, as long as it leads to a correct answer.

All experiments are run for all model sizes. We expect the larger models to gain more performance from the more refined settings. The above listing is in order of expected performance gain. However, for the smallest model with 560m parameters, we do not expect a lot of improvement across the different settings.

4.3 Hyperparameter settings

For all models and datasets, we used a learning rate of 10^{-4}. We train with a batch size of 4 for up to 50 epochs with early stopping applied. The rank of the LoRA was set to 8, using LoRA alpha of 32, and drop out with a probability of 5%. We use no bias for LoRA. No repetition penalty or beams were used during training.

5 Results and analysis

5.1 Few-shot baselines

Figure 1 shows the performance of various sizes of Bloom models from 560m to 3B parameters on 3-digit-division, an arithmetic reasoning task, and NLI, a common sense reasoning task.

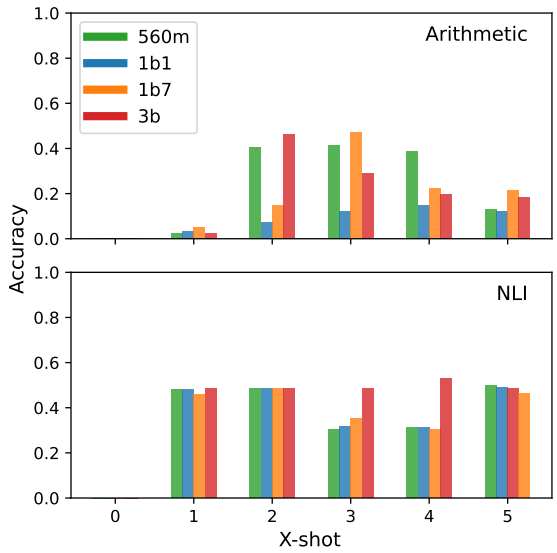

Figure 1: Model performance on 3-digit-division (arithmetic) and NLI in the classical few-shot setting. The length of the support has inconsistent effects on differently sized models for the two tasks. In the zero-shot setting, all models have 0% accuracy on either task.

We observe that without any support, the models' accuracies are at 0% for both tasks. We suppose that the models have not been (sufficiently) trained on this task. Therefore, it constitutes a novel task that the model cannot solve without in-context learning. This is supported by the increase in performance across all models when initially increasing the size of the support. Whereas a single question-answer example only minimally increases the models' accuracy for 3-digit-division, the 2-shot setting already provides a substantial increase in performance for the smallest and largest model.

Arithmetic resoning

For the 3-digit-division dataset, differently sized models tend to respond differently to the length of the support. The behavior seems almost random. Unexpectedly, the smallest model scores among the best performances with supports varying from a length of 2 to 4, but drastically loses performance when faced with 5 examples. Similarly, the largest model performs best with 2 examples and worsens with any other support added. The precise reasons for this behavior are not fully clear to us and fall out of the scope of this research project. Further investigation might be necessary to explain this behavior.

NLI

Similarly to what we observed for 3-digit-division, the models have 0% accuracy in the zero-shot setting on NLI, presumably for a lack of training on this task. However, in this task, we see a much more consistent behavior across the different few-shot settings. Generally speaking, the length of the support does not matter too much for the largest Bloom model. The smaller models have a hard time working with the 2-shot and 3-shot settings. A deep dive into this falls out of scope for this project but may constitute an interesting research topic for the future

5.2 CoT and finetuning

Figure 2 shows the performance increase for various sizes of Bloom models from 560m to 3B parameters on 3-digit-division, an arithmetic reasoning task, and NLI, a common sense reasoning task.

Arithmetic reasoning

Comparing the accuracies for the 4-shot and 4-shot CoT settings reveals inconsistency in the response of differently sized models to the addition of CoT to the support. The midsize models profit from an additional explanation in the support. The 3B parameter Bloom model is slightly negatively impacted by them and the 560m parameter model is very negatively impacted by the addition of CoTs. The latter can be explained when considering that

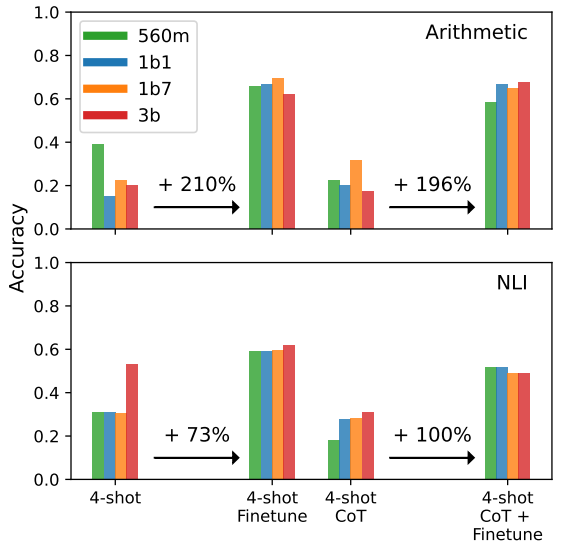

Figure 2: Increase in model performance on 3-digit-division (arithmetic) and NLI for finetuning the models without and with CoT. 4-shot and 4-shot CoT baselines are roughly similar, with the classical setting usually outperforming the CoT setting. Finetuning in both settings yields roughly equal performance increases. The performance increase annotated in the figures is averaged over all 4 models. Note how the 4-shot (i.e. no CoT or finetuning) performance refers to 4-shot performance in Figure 1.

small models are expected to have more difficulties when faced with more complex inputs. Overall, the models do not benefit significantly from CoT prompting without finetuning. This result is in alignment with the findings of Wei et al. (2022), who argue that the model must be of size 100B parameters or larger to benefit from CoT prompting.

When finetuning with CoT supports, we see a significant increase in all models' performance. Our technique helps increase the average performance across all models by 196% This seems to indicate that the models are learning to effectively reason to obtain a better answer. However, considering that normal finetuning is even more beneficial to the models' performance (+210%), we argue that the models do not truly learn to reason but rather gain in performance through finetuning on the task itself. Furthermore, through a random qualitative sample survey, we perceive that the models' explanation quality does not correlate with the correctness of the answer. This further supports the idea that the models do not learn to reason properly with this technique.

NLI

Comparing the accuracies for the 4-shot and 4-shot CoT shows that adding CoTs to the support is harmful to some models' performance and has no significant effect on the midsize models. We attribute this negative impact to a specific behavior emerging with CoT prompting. The models start producing an explanation and get lost in that process without ever generating an answer. This will be considered as an incorrect answer, which lowers the average accuracy.

A possible reason why we did not observe this behavior with 3-digit-division is that it benefits more from CoT reasoning than NLI. Hence, the effects cancel out. In support of this theory, we argue that CoT finetuning is also more effective for 3-digit-division than for NLI.

As for 3-digit-division, when finetuning with CoT supports on NLI, we see a significant increase in all models' performance. For NLI this increase is at 100% on average across all models, compared to 196% for 3-digit-division. Further investigating the models' outputs, we observe that our technique helps to alleviate the issue of endless explanations. Nonetheless, we observe that the final performance of finetuning with CoT is still lower than the final performance of finetuning without CoT, even though the performance increase was lower for the latter technique (+73%).

Again, we must attribute most of the performance increase to finetuning on the dataset, instead of to the models learning to properly reason about the task. We conclude that 4-shot CoT finetuning does not yield better results than regular 4-shot finetuning.

6 Conclusion

We have explored the effects of CoT finetuning with a small static subset of explanations to question-answer pairs. We observe that our technique does not outperform other finetuning methods that do not use CoTs. These results indicate that smaller language models do not learn to reason without explicitly training on ground truth CoTs. Overall, our technique does not yield the same results as finetuning smaller language models on ground truth CoTs, generated by a larger teacher model (Magister et al., 2022; Kim et al., 2023; Ho et al., 2022).

Since the models were never trained to give a logically consistent explanation - e.g. by using a

ground truth CoT label with teacher forcing - the quality of an explanation does not correlate with the correctness of an answer. This would be acceptable, in our opinion, as we only want to improve the models' performance on the task. We do not aim to provide a CoT that can explain the model. However, given that this technique performs worse than regular few-shot finetuning, we conclude that improving the models' explanations is relevant to improving its answers. Therefore, future research should tackle this issue and improve on the quality of the models' reasoning.

Furthermore, we conclude that all Bloom models from 560m to 3B parameters benefit from this technique, indicating that the threshold for the minimum model size lies below 560m parameters. This is in accordance with the results of Magister et al. (2022) which show that T5 Base (Raffel et al., 2020) with 220m parameters already benefits from CoT finetuning. Future research could focus more on finding the precise threshold for models available in smaller sizes than Bloom. Nonetheless, having a technique that outperforms regular few-shot finetuning would be a prerequisite for this research.

For future research, we would like to investigate whether we can uncover reasoning capabilities with a similar setup but with a few variable supports. That is, we would still train on answers only but build the support differently at each forward pass. This setting would still operate without a teacher model that produces ground truth CoTs. Furthermore, it may help to reevaluate the design decisions taken for training the model (see section 3). Other settings might improve the technique.

Limitations

Our method does not teach models to reason. Furthermore, it has only been tested on a small subset of reasoning tasks. CoT finetuning in this manner might not be suitable or optimal for other tasks that do not involve reasoning. Even other reasoning tasks might not see benefits from this method, as we have seen some differences in performance increase even across the 2 tasks at hand.

The CoT, generated by the models should never be used as explanations for their answer, since the model was never trained on ground truth CoTs. Its sole purpose is to improve the answer, even if the reasoning is completely logically wrong.

Ethics statement

While stronger and more capable language models always come with more of the classical ethical issues - such as hallucinations, social biases, etc. - we do not believe that our method adds any new ethical dilemmas to this. Nonetheless, misusing the method for the purpose of explainability can have serious implications. Our method does not require a correct explanation to arrive at a correct answer. Therefore, the CoT should never be used as an explanation for the model's decision.

Acknowledgements

We would like to thank Konstantinos Papakostas for his support during this project.

Submission details

Code

The code developed for this project can be found at `https://github.com/elidub/CoT`.

Contributions

Robin: Creating the poster and the report (wrote all sections that are not explicitly mentioned by the other members). Designing the different ablations, i.e. which settings should be evaluated. Exploring the use of PEFT (LoRA) and how to finetune the model. Working on a method to extract the answer from the explanation (dead-end, was discarded). Qualitative testing of pretrained models' performance for model selection.
Kieron: Co-writing the dataset structure. Performing background research on the different tasks of BigBench. Working on the the custom loss computation for finetuning with explanations, parametrizing the script and tuning hyperparameters. Keeping track of all experiment runs. Gathering all experiment results and performed the initial analysis. Writing Section 3.3.
Orestis: Performing background research for BigBench and the CoTs by Lampinen et al. (2022). Selecting the tasks used and implementation of the dataset for each task. Extracting the CoTs and combining them with the original BigBench dataset. Working on the trainer for the few-shot baselines. Working on datapoint structure for each experiment (0-shot, few-shot, CoTs, etc).
Elias: Exploring which LLMs can be used and enabling them to be stored on Lisa. Setting up backbone of training pipeline by adapting existing

pipelines. Developing custom loss function: extracting only a from the the model's output that includes explanations without using for-loops, writing the final trainer for the few-shot baselines. Preparing the dataset used for the baselines. Writing Section 4.2. Creating all figures.

Jonathan: Working on various aspects of codebase including: the main training loop, the custom loss computation for finetuning with explanations, integration of dataset with training setup, debugging of models and training setup. Working on initial exploration of PEFT model training setup.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020a. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation.

Seungone Kim, Se June Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. 2023. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.14045*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Andrew K Lampinen, Ishita Dasgupta, Stephanie CY Chan, Kory Matthewson, Michael Henry Tessler, Antonia Creswell, James L McClelland, Jane X Wang, and Felix Hill. 2022. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.