# Enhancing Large Language Models' Reasoning Capabilities through Contrastive Prompting and Direct Preference Optimization

Altria Wang
*Department of Computer Science*
*Columbia University*
New York, NY USA
altria.wang@columbia.edu

Kevin Tang
*Department of Computer Science*
*Columbia University*
New York, NY USA
kaiwen.tang@columbia.edu

***Abstract*—Large language models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation, yet their reasoning abilities often remain inconsistent, particularly in tasks requiring logical coherence or multi-step problem-solving. Several existing approaches, such as RLHF or targeted fine-tuning, can improve the model's general conversational and reasoning capabilities but also introduced significant implementation cost. This project explores the potential of using Contrastive Prompting (CP) and Direct Preference Optimization (DPO) to create a self-supervised solution to enhance reasoning performance in LLMs while reducing the implementation complexity and total cost. Our result indicates that the accuracy post DPO is bounded by baseline accuracy from the contrastive prompting. GitHub Repository: https://github.com/KevinTang2318/cp_dpo**

## I. Introduction

In recent years, Large Language Models (LLMs) have shown outstanding capabilities in natural language understanding and generation. The presence of LLMs in the industry is also enhanced as they are deployed for several downstream tasks. However, despite this increased significance, LLMs' performance are known to be unstable on solving complicated tasks that require reasoning. To enhance the performance of LLMs, techniques such as Reinforcement Learning with Human Feedback (RLHF) and task-specific fine-tuning are often applied to align and fine-tune the base LLMs for better response quality and increased conversational and reasoning capabilities. While effective, these methods can be resource-intensive, requiring human annotation, extensive data collection, and further training. Such side effects made this fine-tuning method difficult to scale, prone to human bias, and require extensive computational resources.

Methods such as Reinforcement Learning from AI Feedback (RLAIF) were proposed to attempt mitigating the human labor cost involved in RLHF. While effective, RLAIF preserved problems such as complicated implementation, resource intensive training process, and the introduction of bias from the AI model [6]. As the research in relevant areas progress in recent years, several methods such as Prompt Engineering and variants of preference fine tuning have been proposed to enhance the model's performance. This leads us to the question: with recent advancements, is there a more efficient approach to fine-tune LLMs while achieving performance equal to or exceeding that of existing methods?

In this study, we aim to explore the effectiveness on improving LLM's performance by utilizing the combination of Contrastive Prompting (CP) and Direct Preference Optimization (DPO) to fine-tune an instruction-tuned LLM. Our goal is to improve the LLMs' reasoning capabilities while reducing or eliminating the reliance on human annotators for preference and prompt dataset creation, allowing LLMs to use their own generated responses for fine-tuning and ultimately improving performance. Such a method will offer more accurate AI feedback, easier-to-implement fine-tuning paradigm, and reduce the total computational cost compared to RLHF. We specifically focus on the model's reasoning capability as this is the area where LLMs demonstrated the most inconsistency. We will evaluate the potential improvements in reasoning capabilities using various reasoning datasets, including arithmetic, common sense judgment, symbolic reasoning and other reasoning tasks.

This work aligns with ongoing research trends aimed at developing more efficient and scalable methods for fine-tuning LLMs. By exploring the combination of CP and DPO, we provide insights into the potential of self-supervised, preference-based, resource-efficient fine-tuning. If successful, this approach could pave the way for more robust LLM optimization techniques that generalize across tasks, ultimately enabling the development of AI systems with enhanced reasoning capabilities and reduced computational requirements.

## II. Related Works

### A. Reinforcement Learning from Human Feedback (RLHF)

LLMs are language models trained on an extensive amount of text data. This process is referred to as pre-training. [4] and the trained LLM is referred as base LLM. Despite pre-trained base LLMs have good capabilities on language understanding and generation, their performance on more complicated downstream tasks can still be relatively limited. Additionally,

these models possess the potential to generate harmful or undesirable outputs that may conflict with human intentions. As a result, further fine-tuning and alignment are essential to enhance their performance and ensure their outputs align with human values and expectations.

The technique that was mostly used to align base LLMs with human preferences and to improve its response precision is Reinforcement Learning from Human Feedback (RLHF) [2]. RLHF achieves such a fine-tuning operation by first training a reward model using a human preference dataset, and then use another prompt dataset to fine-tune the base LLM with reinforcement learning, usually with Proximal Policy Optimization (PPO). The human preference dataset is generated by human annotators first ranking different versions of model completions on the same prompts and forming pairwise training samples using their ranks. While such a method can improve the model's performance and achieve excellent alignment with human preferences, it requires extensive human labor to prepare the two datasets. To solve this issue, Reinforcement Learning by AI Feedback (RLAIF) [8] was proposed. Instead of relying on human feedback, RLAIF uses a second language model to generate the preferences, therefore reducing the human labor costs. However, such a method is prone to errors, as the feedback from another AI model may not always align with human values or provide nuanced judgments.

### *B. Direct Preference Optimization (DPO)*

Despite the expensive annotation cost required by RLHF, its training procedure is also complicated and may involve high computational cost, since it involves training a separate reward model, which is often also a language model. The method of Direct Preference Optimization (DPO) [10] was proposed to address this issue. This is a stable, performant, and computationally light algorithm that can achieve near-identical results compared to RLHF [10]. DPO eliminates the need for reinforcement learning algorithms like PPO, which are often complex to implement and tune. Instead, DPO directly optimizes the model parameters by minimizing a loss function derived from the preference data. The loss function compares the likelihood of preferred and non-preferred outputs, encouraging the model to generate responses similar to those marked as "preferred." The preference dataset here is formed by selecting which response the human annotator prefers among the two given model completions, which also eliminates the need of ranking and complicated dataset generation.

### *C. Obtaining more accurate response from LLM using prompt engineering*

Prompt Engineering is an emerging area that has shown outstanding capabilities in improving the LLM responses [1]. Several techniques, including Chain-of-Thought [13] and Tree-of-Thought [16] prompting, have shown their capability of prompting the model to produce more accurate responses and improving its reasoning capability. Despite their successes, few-shot CoT needs human-labeled reasoning steps for each example, while zero-shot CoT may generate incorrect reasoning steps (especially for commonsense and arithmetic reasoning) [15].

To address such issues, Yao et al proposed Constrative Prompting (CP) [15], which is an intuitive approach that prompts the model to generate both correct and incorrect solutions. CP is a two-step procedure: in the first step, the prompt is appended with the message **"Let's give a correct and a wrong answer"** to encourage the model to generate both correct and incorrect solutions. In the second step, the authors employ a few-shot-like approach, prompting the model to summarize the final answer by synthesizing insights from the generated correct and incorrect solutions. Such an approach has shown great potential in improving the model's reasoning capability and can be integrated with other prompting techniques easily.

Based on our review of the above approaches, we believe that combining CP with DPO may lead to a simpler yet effective fine-tuning method to improve the performance of the base LLMs. Compared to RLAIF, such an approach enhanced the precision of AI feedback and also reduced the computational cost required for performing fine-tuning. Additionally, such a self-supervised approach is easy to scale and can be deployed with various model architectures.

## III. Methodology

### *A. Experimental Setup*

We adopted a similar approach to the set up in [15]. The original paper proposed a framework with two step prompting. The model is first asked to provide a correct and incorrect answer to a reasoning task. The correct answer is later fed to the model again to extract the numerical answer to the original question. Instead of answer extraction, we used the correct and incorrect answer as the positive and negative preferences in the DPO finetuning process. See Figure I for an illustration of our experimental setup.

For model, we used Llama3-8b-Instruct [11] for scalability concern. In practice, we observe substantial difference in the base model performance with instruction-tuned model performance in following the task description. In addition, full parameter DPO requires expensive computing resources which may quickly run out of the budget from Google Cloud. We used LoRA [7] to reduce the possibility in getting Out of Memory (OOM) situation [3].

For dataset, in order to test whether DPO can accomplish reasoning task in a better performance through this automatic pipeline, we remained consistent with the original paper used in [15], which includes arithmetic, common sense judgment, symbolic reasoning and other reasoning tasks. For time and computing resource considerations, we planed to use only a selected dataset of the list. We did not include all the datasets from the original paper partly due to limited time constraint, partly because they are not suitable for a DPO setting. Using these datasets brings two benefits. Firstly, we can easily compare the performance with the non-DPO tuned version, i.e only contrastive prompting based performance.

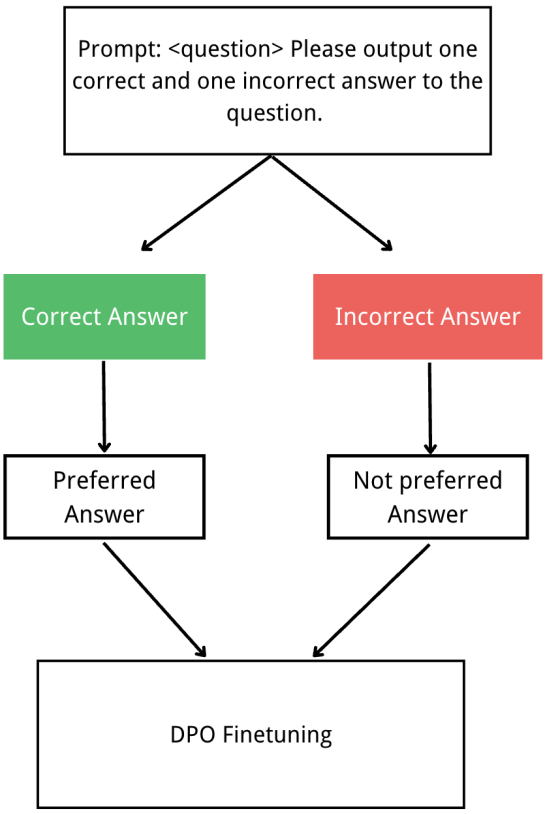


Fig. 1. Graph for experiment

Secondly, these reasoning tasks provide us a ground truth answer, which simplifies later evaluation metrics. If time allows, testing whether using ground truth as the correct label and keep the incorrect answer as the not preferred answer will also be in scope as a gold baseline.

### *B. Data Collection and Preprocessing*

To comprehensively evaluate and enhance the model's reasoning capabilities while effectively showcasing its performance improvements, we selected six datasets spanning four major reasoning tasks. These datasets include tasks where prior studies, such as Yan et al. [15], reported limited performance, providing a basis for comparison and further analysis. The datasets we selected are [5], [9], [12], [17]:

- Arithmetic Reasoning
  - **AQUA**: Multiple Choice
- Common Sense Reasoning
  - **Strategy QA**: Yes/No
- Symbolic Reasoning
  - **Last Letter**: Open-ended
  - **Coin Flip**: Yes/No
- Other Reasoning
  - **BigBench Date**: Multiple Choice
  - **BigBench Object Tracking**: Multiple Choice

To generate the preference datasets, we first perform a train-validation-test split of 70%, 20%, and 10% on each dataset. The training and validation sets are then input to the LLaMA3-8B-Instruct model using the Contrastive Prompting (CP) method to generate responses that include the correct answer, the wrong answer, and the model's reasoning. An example question is presented below:

```
Q: A coin is heads up. Gavin flips the
coin. Neha does not flip the coin. Asha
does not flip the coin. Baltazar does not
flip the coin. Is the coin still heads
up? Note that "flip" here means "reverse".
Answer this question with either "Yes." or
"No." first, then provide your reasoning.
A: Let's give a correct and a wrong
answer. Output your answer in the
following format: "Correct answer: <yes
or no>; Wrong answer: <yes or no>; <your
reasoning>
```

These responses are subsequently processed through a parsing pipeline to extract relevant components and construct the final prompt dataset for CP + DPO fine-tuning. Each sample in the preference dataset consists of the original question, answer options, ground truth, correct answer, wrong answer, and the model's reasoning.

In contrast to the training and validation sets, the test dataset is processed separately. The LLM is prompted using a standard (non-CP) prompt to generate a direct answer. The selected answer option is then extracted using regular expression (Regex) patterns, and a baseline test accuracy is calculated for each dataset.

To keep influence from parameters involved in inference minimum, we set the temperature to be 0.01 and max new token to be 512. Empirically, this setting yield stable output that follows the format we requested in the prompt.

### *C. Training and Optimization Procedures*

We adopted code from publicly available repository [14]. In our GiTHub, we only included the codes we used or modified from [14] and ignored other unrelated files. We reformatted the dataset into the following format, where we substitute the choice rejected and choice chosen to be the correct and incorrect answer from the CP output. For the questions with multiple choices, we only included two of them but kept all options available in the original question.

```
"question": "Q: Would Christopher
Hitchens be very unlikely to engage in
tonsure? Answer this question with either
"Yes." or "No.","
"response chosen": "Yes",
"response rejected": "No",
"ground truth": "Yes"
```

We trained the LoRA with DPO method on NVIDIA GeForce RTX 3090s. Qualitatively, we saw repetition and

non-sense or non related sentences in sequence generated. We suspected this could be due to catastrophic forgetting of the task (namely language modeling) after finetuning, which may be a phenomenon to overfitting. Thus, we tested multiple hyperparameter combinations involved in LoRA training and settled with rank = 4 and alpha = 8 with dropout rate = 0.1. In addition, we changed the target training module to be only the q matrix, k matrix, v matrix compared with full modules (which includes downward projection, gate projection, upward projection, and o matrix). We found a significant increase in model output quality with the current setting compared with other parameter configurations (such as with rank = 8, alpha = 16).

The total parameter trainable parameter count is $10485760$ (out of $8040747008$), which roughly translates to $0.14\%$. The training lasted 3.2 epochs and reached an evaluation loss of 0.0512.

## IV. Results

Parallel to Yao et al., we use the accuracy on the selected reasoning datasets to evaluate the model's reasoning capability. [15] Table I demonstrates the comparison between the model's performance before and after fine-tuning using CP + DPO.

| | Zero-Shot (baseline) | Zero-Shot-CP-DPO |
|---|---|---|
| AQuA | 0.39 | 0.38 |
| Strategy QA | 0.66 | 0.63 |
| Coin Flip | 0.46 | 0.46 |
| Last Letter | 0.14 | 0.02 |
| BigBench Date | 0.43 | 0.38 |
| Object Tracking | 0.29 | 0.27 |

TABLE I
ACCURACY COMPARISON BETWEEN ZERO-SHOT PROMPTING USING STANDARD LLAMA3-8B-INSTRUCT AND ZERO-SHOT PROMPTING WITH LLAMA3-8B-INSTRUCT FINE-TUNED USING CP + DPO

We compare the difference in the baseline Zero-shot prompting and Zero-shot result after training with DPO method from CP method. Unfortunately, we did not find much significance in the performance. The performance pairs are mostly very similar to each other except for the Last Letter pair. In the Last Letter finetuning, both preferred answer and not preferred answer are purely letter combinations (such as "aabb"). This means that model face great difficulty in inferring the intrinsic pattern of the letters from the answers.

To comprehensively analyze the performance changes introduced by fine-tuning with CP+DPO, we conducted a precision analysis on the preference datasets generated using the CP approach on the LLaMA3-8B-Instruct model. Specifically, we measured the precision for correct answers (where the generated response matches the ground truth) and incorrect answers (where the generated response deviates from the ground truth). The results of this analysis are presented in Table II and Table III.

We can derive several insights from the precision comparisons. First, the precision for the generated correct answers is relatively low across all tasks, with the highest score of 0.63 observed on the Strategy QA training set (assume we only compare training sets here). Notably, the model performed extremely poorly on the Last Letter task, achieving a correct precision close to zero, which indicates that very few questions were interpreted or solved correctly. Additionally, the model demonstrated limited capabilities in solving mathematical problems. All such undesired behaviors likely due to the constrained model size of LLaMA3-8B-Instruct, since the experiment result shown by Yan et al. are relatively higher when using GPT-4. [15]

| | Correct Precision | Wrong Precision |
|---|---|---|
| AQuA | 0.29 | 0.77 |
| Strategy QA | 0.63 | 0.63 |
| Coin Flip | 0.46 | 0.46 |
| Last Letter | 0.01 | 0.99 |
| BigBench Date | 0.50 | 0.92 |
| Object Tracking | 0.30 | 0.67 |

TABLE II
PRECISION COMPARISON BETWEEN TRAINING PREFERENCE DATASETS

| | Correct Precision | Wrong Precision |
|---|---|---|
| AQuA | 0.27 | 0.82 |
| Strategy QA | 0.66 | 0.66 |
| Coin Flip | 0.46 | 0.46 |
| Last Letter | 0.01 | 1.00 |
| BigBench Date | 0.45 | 0.97 |
| Object Tracking | 0.28 | 0.64 |

TABLE III
PRECISION COMPARISON BETWEEN VALIDATION PREFERENCE DATASETS

The relatively low precision on correct responses suggests that fine-tuning with such data may mislead the model by reinforcing suboptimal patterns, potentially degrading its overall performance. However, a key observation is the consistency between Table II (training precision) and Table III (validation precision), which indicates minimal distributional differences between the training and validation datasets. This stability could contribute to improved generalization and better performance of the fine-tuned model.

| | Reason - Answer Prompting |
|---|---|
| AQuA | 0.38 |
| Strategy QA | 0.68 |
| Coin Flip | 0.52 |
| Last Letter | 0.42 |
| BigBench Date | 0.73 |
| Object Tracking | 0.41 |

TABLE IV
PRECISION COMPARISON BETWEEN VALIDATION PREFERENCE DATASETS. ALL SCORES ARE SIGNIFICANTLY HIGHER THAN ZERO-SHOT (BASELINE) OR ZERO-SHOT-CP-DPO.

## V. Discussion

The result yield several points and we discuss them one by one here.

### A. Performance depends on the order of output

During inference time after DPO, we used the following prompt using one example from the last letter task:

```
Q: Take the last letters of each
word in "Kristen Herbert Benny El"
and concatenate them. Output your
answer in the following format: Correct
answer: <your answer>; Reasoning: <your
reasoning>.
```

As we can observe, we explicitly asked the model to output the answer first, then provide its reasoning. The same order was used in the contrastive prompting. By eyeballing the output for the last letter task, which is the task we achieved the least performance. The thought process actually has the correct answer, but it differ from the actual answer it outputted in the very start. This suggested another possibility that switching the order of <reasoning> and <answer> could simply improve the performance. We used the model after CP+DPO to test this hypothesis and report the scores in Table IV.

From Table IV, we see an increase in performance in all tasks compared to baseline and CP+DPO by a significant amount. This suggests that any reasoning prompt is highly sensitive to the order of steps. We open the question of whether CP+DPO could work under different sets of prompt (such as asking for reasoning first then giving answer).

### *B. DPO do not improve reasoning in particular given training data with limited accuracy*

DPO is particularly designed to fit human preference when there are two options with more or less similar content available. It's not particularly designed to improve the performance of accuracy or other similar metrics. We test the possibility of automatically feeding outputs from contrastive prompting to DPO, but we do find the DPO results are bounded by the baseline performance. This result is not surprising. If the maximum performance we can reach is baseline, then the model is unlikely to exceed the baseline. Our work suggests that the method DPO itself does not change the reasoning process inherently and the reasoning ability of the model mostly relies on the training data.

We did not find a performance gap between multiple choice question, yes/no (binary choice) question. Open-ended question seemed to be harder than the rest of the question and remained to be the only dataset with significant performance drop.

In addition, we find it hard to control between under fitting and over fitting. With settings that would include more parameters, the model starts to output repetitive sentences that did not make sense. The current setting guarantees a reasonable output, but we are not sure if they made sure the DPO had its maximum effect on the current model.

### *C. Limitation and Future Work*

Although our work did not yield statistically significant results, it suggests several promising directions for future research.

Firstly, researchers could experiment with reversing the sequence order to generate an enhanced training set and replicate our experimental setup. One could also try to use the ground truth as the preferred answer, and automate the not preferred answer extraction through large language model.

Secondly, this framework could be extended to model distillation by leveraging more powerful models like GPT series or LLAMA-70B-INSTRUCT to generate correct and incorrect answers, which could then be used in DPO training for smaller models to achieve automated training.

Thirdly, future investigations should focus on optimizing parameter combinations to better balance underfitting and overfitting through tools like Weights & Biases. Key parameters such as the maximum number of steps (affecting epoch count) warrant further exploration, as they may significantly impact model performance. We also recommend testing various prompting strategies and expanding to larger datasets to overcome our current dataset's size limitations.

## VI. CONCLUSION

In this project, we targeted to improve the model's reasoning capabilities by proposing a novel self-supervised fine-tuning approach. We explored the combination of Contrastive Prompting with Direct Preference Optimization to enhance the reasoning capability of a Llama3-8B-instruct model while reducing the need for human annotation and lowering the computational complexity of fine-tuning. The LoRA fine-tuned model was tested on six different reasoning datasets, covering arithmetic, commonsense, symbolic, and other reasoning tasks.

Although the test results did not demonstrate significant improvements using this method, our analysis revealed that the final model's performance is highly impacted by inaccuracies in the generated responses when leveraging Contrastive Prompting. Additionally, we observed that the order of outputs plays a critical role in determining the precision of the generated responses.

Future work will focus on addressing these limitations. Specifically, we plan to improve the precision of the generated responses by utilizing a larger language model and refining the contrastive prompts further. Additionally, enhancing the accuracy of the fine-tuned model could involve increasing the training data size and, where computational resources allow, removing the constraints of LoRA to perform full fine-tuning. Such efforts will not only improve the model's reasoning performance but also provide deeper insights into the interaction between Contrastive Prompting and Direct Preference Optimization for reasoning tasks, and to draw better insight on whether such a self-supervised fine-tuning method is possible. Based on the results seen in this analysis, we remain positive on the possibility of success for such a supervised fine-tuning method and its prospects.

## REFERENCES


[1] Xavier Amatriain. Prompt design and engineering: Introduction and advanced methods, 2024.

[2] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.

[3] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.

[4] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[5] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies, 2021.

[6] Samuel Höglund and Josef Khedri. Comparison between rlhf and rlaif in fine-tuning a large language model, 2023.

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

[8] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback, 2024.

[9] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*, 2017.

[10] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024.

[11] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[12] Jianing Wang, Qiushi Sun, Xiang Li, and Ming Gao. Boosting language models reasoning with chain-of-knowledge prompting, 2024.

[13] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[14] Ming Xu. Medicalgpt: Training medical gpt model. https://github.com/shibing624/MedicalGPT, 2023.

[15] Liang Yao. Large language models are contrastive reasoners, 2024.

[16] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

[17] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models, 2023.