



THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e. g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

Improving Complex Reasoning in Large Language Models

Yao Fu

Doctor of Philosophy
Institute for Language, Cognition and Computation
School of Informatics
The University of Edinburgh
2025

Abstract

This thesis studies complex reasoning in language models. We use the term *reasoning* to refer to tasks that would require a human to perform slow deliberate, step-by-step thinking (instead of providing an intuitive and instantaneous response), such as mathematical and scientific reasoning, commonsense reasoning, logical reasoning, and strategic reasoning. We use *reasoning capability* to collectively refer to the ability to solve tasks requiring complex sub-problem decomposition and detailed step-by-step analysis.

Our motivation for studying reasoning in language models stems from intriguing theoretical properties (e.g., how scaling laws relate to emergent abilities) and their vast application potential. From an application perspective, we envisage large language models (LLMs) to become the next-generation computational platforms, just like operating systems, and aim to build a new application ecosystem upon LLMs. This vision naturally requires the underlying base model to be able to reason over various complex real-world scenarios. From a modeling perspective, complex reasoning is viewed as a typical ability that emerges with scaling: given other conditions being proper (e.g., given clean data and stable training process), the more compute one spends, the more likely the model has stronger reasoning capability.

We start by reviewing the learning paradigms of large language models, and then discuss fundamental methods for improving reasoning along multiple stages of the model development pipeline. Typically, modern language model development consists of four stages: pretraining, instruction finetuning, reinforcement learning from human feedback, and in-context learning after model deployment. This thesis discusses improving reasoning by in-context learning, finetuning, and learning from feedback. For in-context learning, we propose complexity-based prompting, and demonstrate that the model’s scientific and logical reasoning performance consistently improves as the complexity of in-context demonstrations improves. This work achieved state-of-the-art performance on the GSM8K [Cobbe et al., 2021] and MATH [Hendrycks et al.] datasets at the time it was proposed and has influenced follow-on work by highlighting the importance of data complexity. For instruction tuning, we devise a detailed recipe for specializing smaller language models on mathematical reasoning tasks. We highlight the importance of chain-of-thought formatted data, the use of a finetuned checkpoint, and the balance between capabilities of different directions. This work significantly improved small models’ GSM8K and other math performance by the time it was proposed and has consistently influenced follow-on work by highlighting the importance of capa-

bility balancing. For learning from AI feedback, we show the possibility of constructing a self-improving agent on strategic reasoning tasks by letting agents play against and criticize each other, and show that the ability to self-improve is strongly correlated with the base model and how much it aligns with human instructions. Finally, we review the current state-of-the-art models, highlighting the benchmark saturation problem and the importance of constructing new challenging datasets. We further discuss future directions on multimodal scaling and iterative learning from human, environment, and AI feedback.

Acknowledgements

Many thanks to my advisor Professor Mirella Lapata at Edinburgh for her consistent guidance and support along my PhD journey. Mirella introduced me to three important directions that I am consistently thinking about and studying till today: (1) reasoning and compositional generalization: when a language model is only trained on certain compositions of phrases and sentences, how can it generalize to novel composition; (2) long-context query-focused question-answering: how can language models understand long-context and reason over it; (3) multimodal reasoning: how can language models understand video content and reason over it. My study on machine reasoning composes the major content of this thesis, and I am still consistently studying long-context and multimodal reasoning.

I would also like to thank a lot a friends and researchers that have helped me along the journey. Thank Professor Yansong Feng at Peking University for introducing me to NLP and teaching me how to do research during my undergrad years. Thank Professor John Cunningham at Columbia for introducing me to generative models, and teach me academic writing. Thank Professor Alexander Rush at CornellTech for setting up extremely high standards and introducing me to latent variable models. Thank Dr. Tushar Khot, Dr. Ashish Sabharwal, and Dr. Peter Clark at Allen Institute for AI for introducing machine reasoning to me. Thank Professor Hao Peng at UIUC for consistent help and collaboration. Thank Professor Yoon Kim at MIT, and Dr. Rameswar Panda at MIT-IBM AI Lab for guiding my research on long-context language models. Thank Professor Lei Li at CMU for insightful suggestions at many key decisions in my previous life. Thank Professor Wenhua Chen at Waterloo, and Dr. Xiang Yue at CMU for in-depth discussions on reasoning and long-context. Thank Dr. Xuemei Gu and Dr. Kai-Fu Lee for their generous support of my research on long-context language modeling. Thank Dr. Mahesh Sathiamoorthy for introducing me to my team at Google DeepMind.

And most importantly, thank my family, for their consistent and unconditional support.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Yao Fu)

To my high school physics teacher, Zixia Wang, who passed away due to nasal and sinus cancer: You taught me that the pursuit of knowledge transcends power, wealth, fame, or social status. It's about the pure, unadulterated joy of uncovering the mysteries of the unknown. Your legacy lives on in every discovery I make.

Table of Contents

1	Introduction	1
2	Preliminaries	5
2.1	Roadmap for Building Large Language Models	6
2.1.1	The Decoder-Only Transformer Architecture	7
2.1.2	Pretraining	7
2.1.3	Instruction Finetuning	9
2.1.4	Learning from Human, Environment, and AI feedback	9
2.1.5	In-Context Learning	10
2.2	An Overview of Machine Reasoning	11
2.2.1	Typical Reasoning Tasks	12
2.2.2	Initial Efforts before Large Language Models	13
2.2.3	Large Language Models Unify the Paradigm	14
2.3	Reasoning with Large Language Models	15
2.3.1	Prompting	15
2.3.2	Evaluation	16
2.3.3	Our Contribution	17
3	Improving Reasoning by In-Context Learning	19
3.1	Introduction	20
3.2	Related Work on In-Context Learning	21
3.3	Complexity-based Prompting	23
3.3.1	Selecting Complex Samples as Prompts	23
3.3.2	Complexity-Based Consistency	24
3.4	Experiments	25
3.4.1	Experimental Settings	25
3.4.2	Main Results	28

3.4.3	Analysis	31
3.5	Summary and Follow-on Work	34
4	Improving Reasoning by Instruction Finetuning	37
4.1	Introduction	38
4.2	Background	40
4.3	Specializing Multi-Step Reasoning	42
4.4	Experiments	44
4.4.1	Overall Performance Tradeoff	44
4.4.2	Scaling Behavior of Smaller Models' CoT Ability	48
4.4.3	Specialization Process and Generalization Behaviors	50
4.4.4	Further Design Choices Analysis	51
4.5	Summary and Follow-on Work	53
5	Improving Reasoning by Learning from Feedback	55
5.1	Introduction	57
5.2	Problem Setting	59
5.3	Related Work	61
5.4	Experiments	63
5.4.1	Experiment Setup	63
5.4.2	Base Experiments	65
5.4.3	Behaviors of Different LLMs	67
5.4.4	Towards Continuous Improvements from Iterative AI Feedback	68
5.5	Summary and Follow-on Work	70
6	The Future of Machine Reasoning	73
6.1	Current State of the Art Models	75
6.2	Future Directions	77
7	Conclusions	81
	Bibliography	83

Chapter 1

Introduction

Starting from the release of ChatGPT [OpenAI, 2022] in December 2022, large language models (LLMs) have impressed the world with their generic abilities to perform a spectrum of tasks. LLMs exhibit a wide range of important capabilities, such as in-context learning [Brown et al., 2020b], zero-shot generalization [Mishra et al., 2022], instruction following [Ouyang et al., 2022], coding [Chen et al., 2021b], dialog [Thopilan et al., 2022], and so on. Among this spectrum, complex reasoning lies at the core position, as reasoning, the ability to perform tasks that require slow deliberate thinking [Stanovich and West, 2000, Goyal and Bengio, 2022], is viewed by a large portion of the community [Wei et al., 2022b, Huang and Chang, 2023, Wang et al., 2023b] as a fundamental capability that supports various higher-level applications. We aim to improve various types of reasoning in general, and use mathematical, scientific, logical and strategic reasoning as our benchmarks to study effective methods for improving reasoning across the model development pipeline.

This thesis features a systematic perspective over multiple stages of model development. Typically, there are four fundamental learning paradigms in developing language models: (1). pretraining [Brown et al., 2020b, Chowdhery et al., 2024], where one trains a large model (e.g., a 70B LLaMA in Touvron et al., 2023) on a large scale of high-quality data (e.g., 2 trillion tokens) to obtain a powerful base model; (2). instruction finetuning [Wang et al., 2022b, Chung et al., 2024], where one uses a diverse set of natural language instructions to finetune the base model and unlock the model’s instruction following and zero-shot generalization capability; (3). learning from human [Ouyang et al., 2022], AI [Bai et al., 2022b], and environment [Liu et al., 2022c] feedback, where one further improves the model’s reasoning, safety, precise instruction following, and other important capabilities using feedback signals. (4). in-context learning [Brown

et al., 2020b], where one constructs instructions and in-context demonstrations to ask the model to perform real-world tasks after model deployment. Since large language models are currently a blooming area, many of the existing studies in these paradigms are still exploratory. Among stages of model development, since pretraining requires a large amount of compute that is significantly beyond academic resources, we consider it outside our scope. This thesis focuses on the key challenges of the other three paradigms: instruction tuning, learning from feedback, and in-context learning. We consider rigorous quantification of the problem space and propose initial methods for improving machine reasoning.

Chapter 2 aims to establish the fundamentals of our framework. We start with a short review of the generic roadmap for building large language models, i.e., pretraining, finetuning, learning from feedback, and in-context learning. We also discuss the instrumental role of the base model, and how other learning paradigms (finetuning, prompting, and learning from feedback) interact with pretraining.

In Chapter 3, we consider prompting methods for complex reasoning. We start with the classical problem of example selection for in-context learning, focusing on effectiveness and consistency. We propose *complexity-based prompting* [Fu et al., 2022b], a simple and effective prompt construction principle for a variety of reasoning tasks, where the key takeaway is to *encourage the model’s deep thinking by selecting complex reasoning paths over simple ones*. We use extensive experimental results to demonstrate its effectiveness.

In Chapter 4, we study finetuning methods for improving reasoning, highlighting data engineering and capability balancing [Fu et al., 2023c]. The key takeaway here is to carefully construct the instruction data, in terms of format, quality and diversity, to improve the target reasoning capability in a generalizable way, without compromising the model’s generic capability. We particularly emphasize the importance of instruction data mixture, which serves as the key to unlocking the model’s zero-shot generalization and maintaining capability balancing.

In Chapter 5, we study AI negotiation under a strategic reasoning game and show the possibility of continuous improvement from self-play and AI feedback [Fu et al., 2023b]. The focus here is to study the important design factors of the environment, such that under the environment we could explore how AI agents can negotiate and improve with minimal human supervision. One key insight is that the capability of continuously improving from AI feedback is likely to be an emergent ability that only exists when models are strong (such that they understand the rule of negotiation) and aligned (such

that they can respond to AI feedback).

Finally, we review the current state-of-the-art models in Chapter 6, discussing how frontier models are saturating even the Olympiad-level challenging datasets, how open-source models are catching up, and the importance of constructing real-world challenging benchmarks. We discuss how the current text-based scaling is likely reaching a limit due to insufficient data, and future research may focus on multimodal scaling and iterative learning from human, environment, and AI feedback.

The work undertaken in this thesis has previously been reported in the following core publications:

- **Yao Fu**, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. *Complexity-Based Prompting for Multi-step Reasoning*. ICLR 2023
- **Yao Fu**, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. *Specializing Smaller Language Models towards Multi-Step Reasoning*. ICML 2023
- **Yao Fu**, Hao Peng, Tushar Khot, and Mirella Lapata. *Improving Language Model Negotiation with Self-Play and In-Context Learning from AI Feedback*. Arxiv 2023

The content of the following publications and technical articles also heavily influenced this thesis:

- Tushar Khot, Harsh Trivedi, Matthew Finlayson, **Yao Fu**, Kyle Richardson, Peter Clark, Ashish Sabharwal. ICLR 2023. *Decomposed prompting: A modular approach for solving complex tasks*.
- Xiang Yue, Xingwei Qu, Ge Zhang, **Yao Fu**, Wenhao Huang, Huan Sun, Yu Su, Wenhui Chen. ICLR 2024. *Mammoth: Building math generalist models through hybrid instruction tuning*.
- **Yao Fu**, Hao Peng and Tushar Khot 2022. *How does GPT Obtain its Ability? Tracing Emergent Abilities of Language Models to their Sources*.
- **Yao Fu** 2023. *Towards Complex Reasoning: the Polaris of Large Language Models*.
- **Yao Fu** 2023. *A Stage Review of Instruction Tuning*.

- **Yao Fu**, Litu Ou, Mingyu Chen and Yuhao Wan. *Chain-of-thought Hub: Measuring LLMs' Reasoning Performance*. ICML 2023, Deployable GenAI Workshop

Chapter 2

Preliminaries

In this chapter, we discuss the background knowledge of the reasoning capabilities of LLMs. We first review the roadmap for building modern large language models (Section 2.1) and note that nowadays language models are built from complicated engineering pipelines, and all stages of the pipeline, collectively contribute to the final model's reasoning performance, thus calling for a systematic study. Then we review the literature on machine reasoning (Section 2.2), discussing how large language models provide a unified solution for reasoning tasks, and identify our contributions along the roadmap (Section 2.3).

Pretraining

↓ **Approach:** pretrain a model of large amount of parameters on large scale high-quality data
Objective: obtain a powerful base model

Instruction Finetuning

↓ **Approach:** fine-tune the base model on diverse fine-tuning data aligned with human intentions
Objective: unlock the models' instruction following and zero-shot generalization capability
Our contribution: a careful study of instruction tuning for multi-step reasoning, highlighting data engineering and capability balancing

Learning from Human / Environment / AI Feedback

↓ **Approach:** further improve the model's alignment by learning from various sources of feedbacks
Objective: improve the models' multiple capabilities, including precise instruction following, safety, reasoning, coding and so on
Our contribution: designing a strategic reasoning game to show language models' potential of continuous improvement from self-play and AI feedback

In-context Learning

Approach: asking the model to perform tasks via few-shot in-context demonstration
Objective: fast adapting language models to various tasks
Our contribution: discovering effective and consistent prompt construction principles for multi-step reasoning tasks

Figure 2.1: The general roadmap for building large language models. We propose solutions for improving language model reasoning via instruction tuning, learning from AI feedback, and in-context learning.

2.1 Roadmap for Building Large Language Models

The development of language models is nowadays very complicated and follows a systematic engineering pipeline [Chowdhery et al., 2024, Brown et al., 2020b, Rae et al., 2021], where the major stages are shown in Figure 2.1. There are four major stages: (1). pretraining, where the objective is to build a strong generic base model; (2). instruction finetuning, where the objective is to *unlock*¹ the model's instruction following and zero-shot generalization capability; (3). learning from feedback, where the objective is to enhance the model's capabilities using human, environment, or AI feedback. These three stages are employed by major commercial models like ChatGPT [OpenAI, 2022]², Claude³, and Gemini⁴. Then (4) in-context learning, where one gives task descriptions and demonstrations for performing that task, is the majority learning paradigm to utilize deployed models to downstream tasks. Below we quickly review each of these model development stages and discuss how they respectively contribute to the model's

¹We use the term *unlock* because we tend to view that these abilities are intrinsically within the base model, rather than injected by instruction tuning.

²<https://chat.openai.com/>

³<https://claude.ai/>

⁴<https://gemini.google.com/app>

reasoning performance.

2.1.1 The Decoder-Only Transformer Architecture

Throughout the thesis, unless specified, we by default consider the influential decoder-only transformer architecture as the model backbone. The transformer architecture, proposed by Vaswani et al. [2017], is a highly parallelizable neural architecture by interleaving multi-head attention, feed-forward, and layer-normalization layers. Later it is improved in Su et al. [2024] (Rotary positional encoding), Zhang and Sennrich [2019](RMSNorm), Zoph et al. [2022] (Mixture-of-Experts), and Ainslie et al. [2023] (Group-Query Attention). When the architecture was initially proposed, it was an encoder-decoder architecture [Sutskever et al., 2014]. Later pretraining works, particularly GPT-3 [Brown et al., 2020b] and PaLM [Chowdhery et al., 2024], discovered that when the model becomes large, the encoder might not be necessary because the decoder itself is powerful enough to encode the input tokens. Moreover, compared to encoder-decoder architecture, decoder-only transformers are easier to scale up on modern accelerators like GPUs using pipeline, tensor, and data parallelism [Smith et al., 2022]. Nowadays, decoder-only transformer is the default architecture for most of the large language models like Llama [Meta, 2024], Mistral [Jiang et al., 2023], Qwen [Bai et al., 2023], Yi [Young et al., 2024], and DeepSeek [DeepSeek-AI et al., 2024].

2.1.2 Pretraining

Pretraining is the most compute-intensive stage of large language model development. There are three key ingredients of pretraining:

- **Data Engineering**, e.g., MassiveText [Rae et al., 2021], RedPajama [Computer, 2023], RefinedWeb [Penedo et al., 2023], Dolma [Soldaini et al., 2023], where one composes a sequence of deduplication and quality filters to construct large scale high-quality data from internet-scale data sources. These datasets typically consist of trillions of tokens from CommonCrawl (webpages), Arxiv (research papers), Github (open-source codes), Textbooks, and many professional sources like law and medicine. This part only requires CPU nodes and thus is achievable with academic-level resources;
- **Supercomputing Infrastructure**, where the key is to build high-bandwidth interconnected AI computing nodes supporting large-scale distributed training (e.g,

Nvidia Selene in [Chen et al., 2021a] and Google Pathways in [Barham et al., 2022]). This stage necessitates an industry-scale infrastructure, involving tens of thousands of AI chips like GPUs or TPUs. The details of supercomputing are out of this thesis’s scope, and by default we assume one already has access to at least a small to middle level computer cluster. From a programmer’s perspective, there are two key metrics one may pay attention to: (a). the number of floating point operations per second (flops), which determines the compute power of a single device (e.g., A100 has 312 trillion flops per second for bfloat16 format); (b). the communication bandwidth between devices, which determines the scalability of how many devices can be interconnected (e.g., A100 NVLink has 600 gigabytes per second).

- **Distributed Training**, where one partitions data, model, and optimizer states of grids of devices [Shoeybi et al., 2019, Rajbhandari et al., 2020], and train a large (e.g., more than 100 billion parameters) transformer model (dense model or mixture of experts in [Zoph et al., 2022]) on a large-scale dataset (e.g., more than 2 trillion tokens in [Touvron et al., 2023]). The key techniques include (a). data parallelism [Shallue et al., 2019], where one partitions a batch of data into multiple devices; (b). tensor parallelism [Shoeybi et al., 2019], where one partitions the feedforward layer and the attention layers into different shards and map them to multiple devices; (c). pipeline parallelism [Huang et al., 2019], where one maps shallower and deeper layers into different sets of devices; (d). zero-redundancy optimizer [Rajbhandari et al., 2020], where one shards the optimizer states into different devices. Again we would not go into much details of these methods, but use them as our training infrastructure.

Pretraining is not blindly scaling up the model and data and then hoping for good performance. Contrarily, the impetus for scaling originates from scientific breakthroughs in scaling laws [Kaplan et al., 2020, Hoffmann et al., 2022], demonstrating a predictable relationship between increased computational investment in pretraining and improved validation loss, which further translates to downstream task measures [Hernandez et al., 2021, Yuan et al., 2023, Wei et al., 2023]. These empirical findings and laws suggest that escalating computational resources will reliably yield models with enhanced performance. In many scenarios, such as coding tasks on HumanEval [Chen et al., 2021b], it is possible to forecast model performance by integrating scaling laws from both pretraining and finetuning phases [OpenAI, 2023].

2.1.3 Instruction Finetuning

After pretraining a base model, the second stage of model development is instruction tuning or supervised finetuning (SFT)⁵, where one finetunes the model with a diverse set of instruction-response data pairs to make the model follow human intention. Initial works around instruction finetuning starts from observations on zero-shot generalization on unseen tasks, such as NaturalInstructions [Mishra et al., 2022, Wang et al., 2022b], FLAN [Wei et al., 2021b, Chung et al., 2024], and T0 [Sanh et al., 2021]. Recent works focus on curating high-quality instruction data, combined with chain-of-thought formatted response, i.e., responses written in a step-by-step fashion, rather than directly giving the answer [Wei et al., 2021b, Fu et al., 2023c, Yue et al., 2024a] usually in a dialog context [Zhou et al., 2023, Sun et al., 2024], see Figure 2.2, to further improve language model’s instruction following and reasoning.

A critical feature of instruction tuning is zero-shot generalization: when the diversity of the instructions is large enough, the model will exhibit capabilities to answer unseen questions / instructions [Mishra et al., 2022, Chung et al., 2024, Zhou et al., 2023]. This capability is important for deployment, because users will always come up with novel queries that are not necessarily included in the finetuning dataset. This capability also shifts the evaluation practices: classical machine learning assumes i.i.d. data of training and testing sets, whereas modern language models are mostly tested on distribution shift settings: one finetunes the model on instruction of certain tasks, and evaluates the models’ capability to generalize to related, but unseen tasks [Chung et al., 2024, Fu et al., 2023b, Yue et al., 2024a].

2.1.4 Learning from Human, Environment, and AI feedback

After supervised finetuning, the next stage of model development is reinforcement learning from human feedback (RLHF). RLHF is initially studied in the training of InstructGPT [Ouyang et al., 2022], then widely followed by the training of WebGPT [Nakano et al., 2021], ChatGPT [OpenAI, 2022], GopherCite [Menick et al., 2022], and Sparrow [Glaese et al., 2022]. RLHF can be decomposed into two steps: (1). reward modeling, where one finetunes a reward model using human preference data; (2). policy training, where one uses the reward model to train the policy model initialized from the instruction finetuned checkpoint. These two steps may be repeated multiple rounds

⁵In some literature this stage is called supervised finetuning (SFT) as opposed to the reinforcement learning from human feedback (RLHF) stage [Ouyang et al., 2022, Glaese et al., 2022]. In general, one may approximately view instruction tuning and SFT as similar things but used in different contexts.

after model deployment, as new preference data can be gathered online on a weekly basis, as is discussed in Bai et al. [2022a], Touvron et al. [2023], Xiong et al. [2024], Dong et al. [2024].

The key insight of reinforcement learning, when compared with supervised learning, is to replace the next-word prediction objective to a learned objective encoded in the reward model which eventually can be traced back to the human preference data [Chung, 2023]. This consequently lead to many different model behaviors when compared with the SFT models, such as more detailed responses, refusal of improper queries, and increased level of safety [Fu et al., 2022a]. RLHF usually requires SFT as bootstrapping, because without SFT, the pretrained checkpoint may not generate any reasonable response for the reward model to pick [Touvron et al., 2023]. The reward model also demonstrates scaling behaviors – the larger the reward model is, the better the performance of its corresponding policy model [Gao et al., 2023].

Further studies consider different types of learning algorithms and feedback signals. Recent work replaces the PPO [Schulman et al., 2017] algorithm with best-of-N [Lightman et al., 2023], i.e. to finetune the policy model on the best responses picked by the reward model, out of N responses, or direct preference optimization (DPO, see Rafailov et al., 2023), i.e., to finetune the policy model on a pair of positive and negative responses. Both algorithms are stable and effective alternatives. In addition to human feedback, the feedback from AI [Bai et al., 2022b] and the environment also provide meaningful signals in terms of improving dialog [Bai et al., 2022b], math reasoning [Lightman et al., 2023] and coding [Le et al., 2022].

2.1.5 In-Context Learning

So far we have discussed three stages for model development. Now we turn our focus to in-context learning, which is usually performed after model deployment to swiftly adapt model behavior to downstream tasks. In-context learning is to use instruction and few-shot demonstration to show the model how to do a job, initially developed in GPT-3 [Brown et al., 2020b], later followed by multiple works, especially chain-of-thought [Wei et al., 2022b] which significantly improves reasoning capability, and leads to the discussion of emergent abilities [Wei et al., 2022a] of language models, that is when models become large, they exhibit the ability to perform tasks that were not explicitly trained on, but are present in larger models and not in smaller models. An example of few-shot in-context learning is given in Figure 2.4, and will be discussed in

<p>Q: Asia bought a homecoming dress on sale for \$140. It was originally priced at \$350. What percentage off did she get at the sale?</p> <p>A: Asia saved $\\$350 - \\$140 = \\$210$ on the dress. That means she saved $\\$210 / \\$350 = 0.60$, or 60% off on the dress. The answer is 60</p>	<p>Q: A green plant absorbs light. A frog eats flies. These are both examples of how organisms</p> <p>A. <u>obtain energy</u> B. escape predators C. produce offspring D. excrete waste</p>	<p>Q: A woman is outside with a bucket and a dog. The dog is running around trying to avoid a bath. She...</p> <p>A. rinses the bucket off with soap and blow dry the dog's head B. uses a hose to keep it from getting soapy C. <u>gets the dog wet, then it runs away again</u> D. gets into a bath tub with the dog</p>
Math Reasoning	Scientific Reasoning	Commonsense Reasoning

Figure 2.2: Examples of math, science, and commonsense reasoning. To get correct answers to these questions, the model needs task-specific knowledge. Such knowledge is either explicitly stated in the training data or implicitly assumed, and the model must have the ability to derive the step-by-step solution.

detail later in section 2.3.1.

One important empirical observation of in-context learning and finetuning is that any data that is useful for in-context learning, tends to be useful for finetuning. The chain-of-thought formatted data is a prominent example, where researchers initially prompt the language model using CoT data [Wei et al., 2022b], then community discovered that finetuning the model with CoT formatted data further improves performance [Chung et al., 2024, Fu et al., 2023c]. This observation gives important practical guidance to data engineering: to discover what kind of data is useful for improving the model, one may always start with prompting to identify the data, then use it for finetuning.

2.2 An Overview of Machine Reasoning

Now we discuss machine reasoning and consider using large language models for reasoning tasks. Human reasoning is usually referred to as tasks that require logical and slow deliberate thinking [Leighton and Sternberg, 2003]. These tasks are usually at a certain level of difficulty for an average human to perform. Machine reasoning, although there may not exist consensus of its strict formal definition [Leighton and Sternberg, 2003, Oaksford and Chater, 2007, Johnson-Laird, 1988, Zhao et al., 2023, Duan et al., 2020], usually refers to having programs to perform tasks that are challenging enough for average or even expert level human intelligence.

2.2.1 Typical Reasoning Tasks

Modern machine reasoning literature studies various types of reasoning, such as math [Cobbe et al., 2021], science [Clark et al., 2018], logic [Suzgun et al., 2023, Liu et al., 2021], commonsense [Zellers et al., 2019] and so on. As is shown in Figure 2.2, the first example shows a math reasoning task where the model needs to construct a step-by-step solution, figuring out the transition from the previous steps to the next steps, and solving the calculation within each step. The second example shows a scientific reasoning case, where the model needs to know the background knowledge about green plants and frogs (which can be derived from text book training data), and compose this knowledge to derive a potential solution. The third example shows a commonsense reasoning case, where most of the knowledge is implicit and grounded by real-world experience and the model needs to derive an action that complies with daily experience, which may not necessarily be explicitly specified in the training corpus.

Reasoning is not just a task on its own, but also an indispensable component for generic language tasks, especially when these tasks enter a higher level of difficulty. For example, in retrieval-based language models [Borgeaud et al., 2022, Wu et al., 2021], the research focus is gradually shifting from correctly identifying relevant documents to reasoning over the credibility of the retrieved documents [Yu et al., 2023b, Xu et al., 2024, Chen et al., 2023a]; in long-context understanding, the research focus is shifting from modeling methods to encode long context to reasoning over information from multiple places within the input [Shaham et al., 2023]. In general, as many tasks are lifted by LLMs to a higher level of complexity, they inevitably involve more and more reasoning requirements.

To navigate different types and scenarios of reasoning, we identify two basic prototypes, as is shown in Figure 2.3. The first one is a chain structure, where to solve a complex problem one derives the solutions step by step [Wei et al., 2022b, Fu et al., 2022b]. This structure is commonly observed in tasks of single domain and medium complexity, such as primary-school level math reasoning [Cobbe et al., 2021]. The second prototype is a decomposition structure, where to solve a complex problem one needs to decompose it into multiple subproblems and solve them one by one [Khot et al., 2022b, Patel et al., 2022]. These subproblems may or may not be within the same domain, and may not exhibit dependency. This structure is commonly observed in tasks of relatively high complexity, such as web-browsing [Yao et al., 2022a,b]. Real-world tasks are typically the composition of the two prototypes, where one may observe chain,

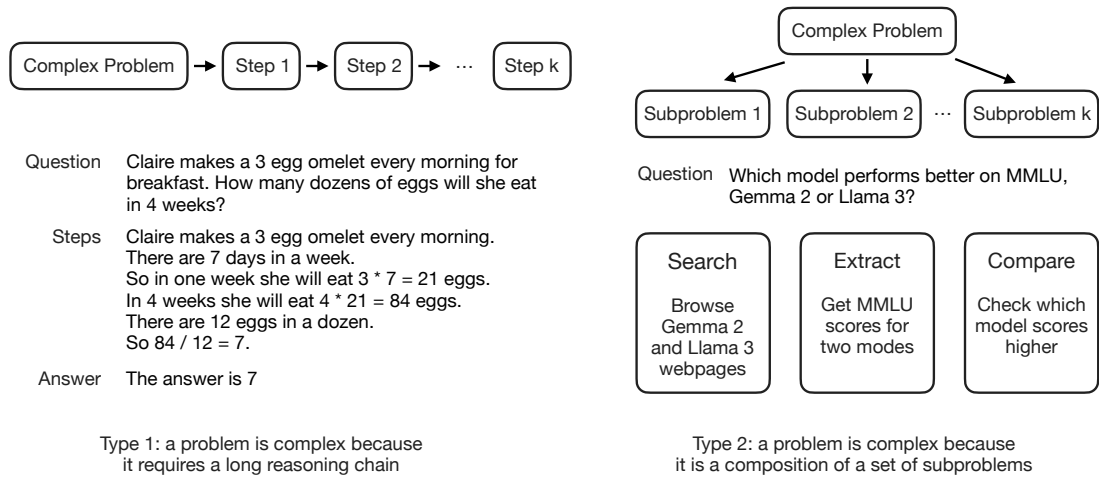


Figure 2.3: Two types of reasoning problems. A reasoning problem is complex because it (1). requires a long chain of intermediate steps [Wei et al., 2022b], or (2). is a composition of a set of subproblems [Khot et al., 2022b]. Real-world complex reasoning, especially when we deploy language models as generalist agents [Xi et al., 2023, Fu et al., 2023b, Shinn et al., 2024], is usually the composition of the two prototypes.

tree [Yao et al., 2023], and directed subgraph structures [Besta et al., 2024].

2.2.2 Initial Efforts before Large Language Models

Before of large language models there was impactful machine reasoning work, which sets the seed and foundation for the recent large language model emergence. Early work studies reasoning over knowledge bases or Wikipedia using statistical methods with linguistic structures [Berant et al., 2013, Fader et al., 2014, Iyyer et al., 2014]. Seo et al. [2017], Yang et al. [2018], Dua et al. [2019] study training neural networks for multi-step reasoning over given contexts. Madaan et al. [2021] studies finetuning language models using various types of rationale and observed promising results. Later Khot et al. [2021], Zhou et al. [2022] study question decomposition and show how complex questions, after decomposition, can be solved effectively because we decompose the difficulty into each subproblem. However, these works suffer from typical problems of small models, such as because the model does not have enough capacity, so even if the question is simpler after decomposition, the model cannot do it; or because the model lacks generality, so although the model can solve questions it is finetuned on, the scope is usually quite narrow, and when there is a slight distribution shift, the model’s performance decreases significantly [Si et al., 2022, Li et al., 2023a]. These drawbacks, largely from the restrictions of small scale, prevent models from being deployed widely.

Large language models significantly improve the limitations of previous work in terms of broader commonsense knowledge, stronger structured reasoning, and better generalization to unseen contexts.

Along the initial study, the theory of compositional generalization serve as a strong driving factor of machine reasoning, where the problem is how a model may generalize to unseen composition of seen items [Finegan-Dollak et al., 2018, Andreas, 2018, Lake and Baroni, 2018]. One example is when the model is trained on counting fruits and classifying cars, can it generalize to count cars and classify fruits. Compositional generalization is believed to be the core capability for the model to reason well, because it assumes the model should apply the rules to combine and utilize its knowledge to derive solutions. In the age of large language models, the difficulty of compositional generalization is lifted significantly to the next level [Drozdo et al., 2022]: from composing common entities using simple phrases like SCAN [Lake and Baroni, 2018] to reasoning over complex dependencies over long document chunks [Xiong et al., 2023], and from composing daily common objects like fruits to composing highly professional objects like medical terminologies [Singhal et al., 2023].

2.2.3 Large Language Models Unify the Paradigm

The emergence of large language models introduces a significant paradigm shift in the machine reasoning literature [Sun et al., 2022, Wei et al., 2022a]. Large language models, as strong reasoning engines, solve problems that previous work cannot solve, and lift the research to the next level. For example, small models may not understand commonsense (like one week means seven days) while large models can, so large models can help the user analyze their weekly expenses based on daily costs. Previous models are domain-specific, such that the lack of the skill of a particular domain may become the blocker of a reasoning chain (e.g., if the user's query involves a three-digit multiplication that the model cannot do) [Khot et al., 2022a, Trivedi et al., 2022]. Now as the model becomes generic, it naturally has a large spectrum of skills (e.g., its calculation becomes more accurate), and if not, it can be trained to call tools (e.g., calling a calculator) [Schick et al., 2023, Cai et al., 2024].

More importantly, large language models open the possibilities of reasoning as deployed agents [Fu et al., 2022b, Yao et al., 2022b,a] in much more complicated games and environments. When equipped with tools like retrieval, web-browsing, API calling, and Python interpreter, we may foster a brand new ecosystem where agents reason

<p>Instruction You are a patient math teacher, your task is to help me solve some math problems. Give me the answer to the following question</p> <p>Question Q: If $x = 2$ and $y = 5$, then what is the value of $(x^4 + 2y^2) / 6$?</p> <p>A. An example about zero-shot instruction</p>	<p>Instruction You are a patient math teacher, your task is to help me solve some math problems. Below are two example math problems and their answers that you may refer to</p> <p>Few shot in-context examples</p> <p>Q: Three positive integers have a sum of 72 and are in the ratio 1:3:4. What is the least of these three integers?</p> <p>A: Let the smallest integer be x. Then the others are $3x$ and $4x$, and the sum of the three is $8x$. So $x = 72 / 8 = 9$</p> <p>Q: Find the value of the first term in the geometric sequence $a, b, c, 32, 64$</p> <p>A: The common ratio is $64 / 32 = 2$. Therefore, the first term is $32 / 2^3 = 4$</p> <p>Question Give me the answer to the following question</p> <p>Q: If $x = 2$ and $y = 5$, then what is the value of $(x^4 + 2y^2) / 6$?</p> <p>B. An example about few-shot instruction</p>
--	--

Figure 2.4: **A.** Zero-shot instruction directly gives the model task description. **B.** Few-shot in-context learning not only gives the task description, but also provides some examples to on-the-fly teach the model how to reason.

in various real scenarios and learn from feedback from humans [Ouyang et al., 2022], the environment [Menick et al., 2022], and other AI agents [Bai et al., 2022b]. This not only enables tremendous opportunities in building AI-native applications, but also lead to the possibility of continuous model improvements by exploring the environment.

2.3 Reasoning with Large Language Models

Now we discuss how to use large language models to perform reasoning tasks, and methods to improve their performance. We first consider when one already has a language model, how to prompt it to reason. Then we discuss methods for evaluating language models' reasoning capability. Finally, we list our contribution to improving language models' reasoning, which will be detailed in the following chapters.

2.3.1 Prompting

Zero-shot Instruction [Ouyang et al., 2022] The most straightforward way of asking a language model to perform a reasoning task is to instruct it to do so. Figure 2.4A gives a math word problem example, where the user first instructs the model to solve math problems, and then asks the question. Nowadays zero-shot instruction is the most ubiquitous way of interacting with the model.

Few-shot In-Context Learning [Brown et al., 2020a] For some hard tasks and tasks that are better explained via examples, one can provide in-context demonstrations in addition to the instruction, for models to better understand the solve the task (Figure 2.4B).

Whether using zero-shot or few-shot depends on two major factors: (1) whether the model is a base model or a chat model: base models are not finetuned to follow instructions yet, so few-shot in-context learning is the most common way of using it; chat models are usually after instruction tuning, so one can use either zero-shot or few-shot (usually few-shot performs better because of the use of demonstrations); (2) the difficulty of the task: usually, if a task is relatively simple (e.g., sentiment classification), then zero-shot should be sufficient; if a task is relatively hard (e.g., teach models to compose Python library functions into larger programs), few-shot prompting is better for fully releasing the model’s capability.

2.3.2 Evaluation

The most common evaluation practice is to use the final answer accuracy as the metric [Cobbe et al., 2021], under the hypothesis that as long as the final answer is correct, the intermediate steps cannot be too wrong. Recent research looks into the intermediate reasoning steps [Lightman et al., 2023], and shows that improving step-wise accuracy, though being much more expensive, is a more effective way of improving reasoning, particularly for hard problems. Furthermore, human preference also plays an important role in guiding the model’s reasoning behavior [Joshi et al., 2023]. For example, among multiple ways of solving a complex problem, human usually prefers the simplest and most straightforward one. Below we explain these three evaluation methods one by one.

Final Answer Accuracy In many cases, evaluating reasoning problems is relatively easy, usually one can use the accuracy of the final answer prediction, typically a number, a phrase, or a choice, as the metrics, as is shown in Figure 2.2. Sometimes the model prediction may not be an exact string matching to the reference answer, for example, the answer is an upper case “A” while the model predicts lower case “a”. Such differences may cause a small level of fluctuation but can be generally stabilized by regular expressions [Fu et al., 2023a].

Step-wise Accuracy focuses not only on the final answer correctness but also on whether the models can get the intermediate steps correct. Improving intermediate

From a systematic perspective, All stages of model development contribute to improvements of reasoning capability

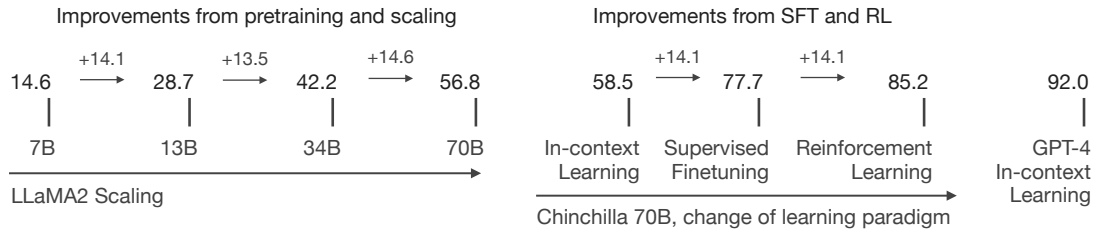


Figure 2.5: Decomposition of performance improvements on GSM8k dataset.

reasoning has been proven to be effective for further boosting model performance on challenging problems [Lightman et al., 2023]. The difficulty here is that, unlike the final answer accuracy which can be easily verified by simple regular expression, verifying stepwise accuracy, as well as training a stepwise reward model, requires much more expensive human annotation. Automating accurate stepwise accuracy with less human annotation [Wang et al., 2023a] is currently an important research direction.

Human Preference When an actual user asks a reasoning problem, say math, the user not only wants to get the final answer, but also expects the model to help them understand how to solve the problem. There are multiple dimensions of user preferences, for example, among different methods to solve the problem, the user usually prefers the simplest solution. Also among different explanation styles (professional, casual, balanced ,etc), a typical user usually prefers a relatively casual tone but without losing professionalism. Annotating and evaluating human preference requires large-scale consistent efforts, and is currently an active research direction [Lambert et al., 2024].

2.3.3 Our Contribution

We note that all stages of model development (pretraining, finetuning, learning from feedback) contribute to performance improvements. One example is shown in Figure 2.5, about how different learning paradigm collectively improves models' accuracy on the GSM8K [Cobbe et al., 2021] benchmark, step by step. Scaling (pretraining) improves the accuracy from 14.6 to 56.8 [Touvron et al., 2023]. Subsequent SFT and RL improve from 58.5 to 85.2 [Uesato et al., 2022]. These findings suggest that to address the challenges of reasoning, we should adopt a multi-faceted approach that targets different stages of model development, as is the practice of this thesis.

In general, the three stages play different roles in improving reasoning. The base model (the pretrained checkpoint) serves as the foundation for reasoning, and the per-

formance is ultimately upper bounded by the base model’s quality, which is traced back to data composition and model scale. Furthermore, recent work shows an analytical relationship between the model’s in-context learning performance on coding and math, the model’s dev loss after pretraining, and the model’s pretraining compute [OpenAI, 2023, Yuan et al., 2023, Hernandez et al., 2021]. This suggests that it is possible that we directly predict the model’s final reasoning performance before running one single line of code. In Chapter 3, we focus on the effective methods to prompt the base model using complex chain-of-thought, identifying key performance factors and source of improvements [Fu et al., 2022b].

Today’s research focus of supervised finetuning (or instruction tuning) is data engineering: to find the correct format and mixture of instruction data such that the model can not only improve reasoning capabilities on the instructions that it has been trained on, but also generalize to unseen instructions after deployment [Fu et al., 2023c, Yue et al., 2024a, Yu et al., 2024, Ivison et al., 2023]. In Chapter 4, we will carefully study the format and mixture of instructions, and their implications to model generalization for both in-domain and out-of-domain settings.

When the model becomes strong, it becomes harder to use average human annotated data to further improve the model. The learning signal shifts from direct human supervision to deploying the model to real scenarios and letting it to learn from feedback [Bai et al., 2022b]. The source of feedback may not only be from human, but also from the environment and other AI agents. The format of feedback can be either a scalar from a reward model, or directly in natural language [Chen et al., 2024a]. In Chapter 5, we focus on natural language feedback, and study the possibility of models’ continuous self-improve in a negotiation game setting [Fu et al., 2023b] via self-play.

The papers discussed in this thesis fostered multiple follow-up works. In complexity-based prompting [Fu et al., 2022b] we identified the complexity dimension, which becomes the basis of following advanced prompting techniques [Zheng et al., 2023]. In SFT [Fu et al., 2023c] we identified the capability balancing problem, which guides a wide array of follow-up studies on instruction data mixture like [Yue et al., 2024a, Yu et al., 2024]. In learning from AI feedback [Fu et al., 2023b] we identify the conditions for language models’ self-improvement, which later become an active topic in the emerging field of AI agents [Xi et al., 2023].

Chapter 3

Improving Reasoning by In-Context Learning

In this chapter, we study the task of prompting large-scale language models to perform multi-step reasoning. Existing work shows that when prompted with a chain of thoughts (CoT), sequences of short sentences describing intermediate reasoning steps toward a final answer, large language models can generate new reasoning chains and predict answers for new inputs. A central question is which reasoning examples make the most effective prompts. We propose complexity-based prompting, a simple and effective example selection scheme for multi-step reasoning. We show that prompts with higher *reasoning complexity*, i.e., chains with more reasoning steps, achieve substantially better performance on multi-step reasoning tasks over strong baselines. We further extend our complexity-based criteria from prompting (selecting inputs) to decoding (selecting outputs), where we sample multiple reasoning chains from the model, then choose the majority of generated answers from complex reasoning chains (over simple chains). When used to prompt GPT-3.5 series, our approach substantially improves multi-step reasoning accuracy and achieves new state-of-the-art (SOTA) performance on three math benchmarks (GSM8K, MultiArith, and MathQA) and two BigBenchHard tasks (Date Understanding and Penguins), with an average +5.3 and up to +18 accuracy improvements. Compared with existing example selection schemes like manual tuning or retrieval-based selection, selection based on reasoning complexity is intuitive, easy to implement, and annotation-efficient. Further results demonstrate the robustness of performance gains from complex prompts under format perturbation and distribution shift.

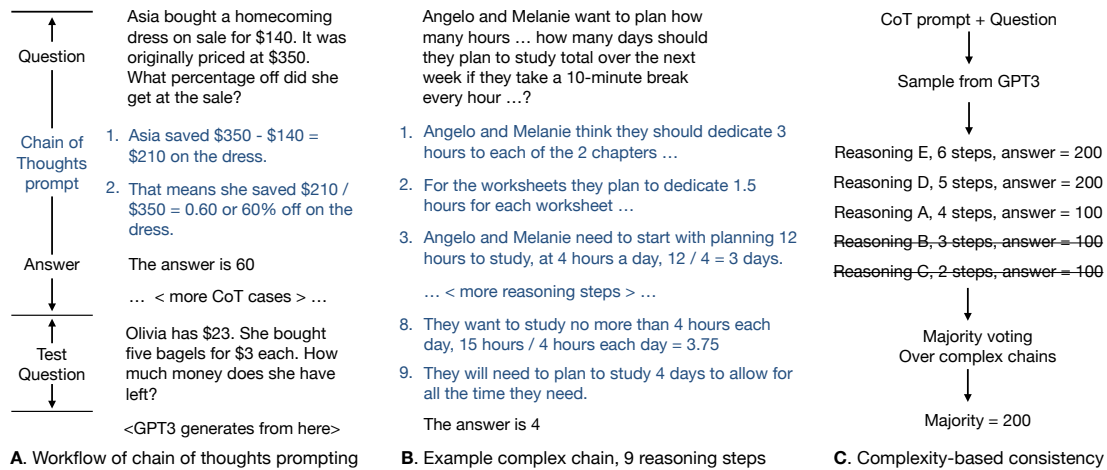


Figure 3.1: **A:** Chain of thoughts (in blue) are intermediate reasoning steps towards a final answer. The input of CoT prompting is a stack of few (often 8) CoT cases before a test question. Then the language model will continue generating an output CoT for the test question. **B:** Chains of harder *reasoning complexity* are chains with *more reasoning steps* (9 steps in this case, v.s. only 2 steps in subfigure A). **C:** During decoding, we sample N reasoning chains from the language model ($N = 5$ here), and take the majority answer over the K ($K = 3$ here) most complex generated chains.

3.1 Introduction

We consider the problem of prompting large language models for multi-step reasoning. Recent breakthroughs [Wei et al., 2022b, Wang et al., 2023b] show that language models, when large enough ($>100\text{B}$ parameters), exhibit the emergent ability [Wei et al., 2022a] of performing complex multi-step reasoning when provided with only a few reasoning examples. In the regime of large models, prompting achieves comparable or even better performance than full training set finetuning while being substantially more sample-efficient [Wei et al., 2022b, Kojima et al., 2022, Lewkowycz et al., 2022]. In particular, Wei et al. [2022b] show that chain-of-thoughts (CoT) prompts, sequences of short sentences describing intermediate reasoning steps towards final answers (Fig. 3.1A), can elicit strong reasoning capabilities from large language models for complex tasks such as math problems.

This work studies *example selection* in chain-of-thoughts multi-step reasoning. Example selection is a central problem in the prompting literature [Liu et al., 2022a, Rubin et al., 2022, Su et al., 2022, Lazaridou et al., 2022]. It asks what instances make the best prompts for solving the tasks of interest. For CoT prompting, example selection is further related to annotation efficiency, as CoT requires manually-annotated reason-

ing chains. For datasets where reasoning annotations are easy to obtain, one may want to know which annotated chains make the best prompt; if the annotations are hard to obtain, one may identify the best cases to annotate, rather than annotating the entire dataset.

We propose *complexity-based prompting*, a new example selection scheme for chain-of-thoughts multi-step reasoning. Existing sample selection methods are usually based on manual tries [Wei et al., 2022b], heuristic rules [Wallace et al., 2019], optimization and search [Shin et al., 2020], or retrieval from a large training set [Rubin et al., 2022]. Different from these schemes, complexity-based prompting chooses examples with complex reasoning chains, i.e., chains with more reasoning steps, as the prompt. Fig. 3.1A shows a simple example with 2 reasoning steps, versus the example in sub-figure B is a complex case with 9 reasoning steps. As we will show in the experiments (section 3.4.2), the reasoning performance of GPT-3 175B [Brown et al., 2020a] clearly improves with the increased input prompt complexity, where complex prompts achieve better performance than simple prompts.

We further extend the complexity-based selection criteria from the input space (the prompts) to the output space (reasoning chains generated by the language model). Our extension is based on the idea of self-consistency [Wang et al., 2023b, 2022a], where they sample multiple reasoning chains (instead of using greedy decoding) from the model that lead to possibly different answers, then choose the majority of the generated answers. Here we propose *complexity-based consistency*, where instead of taking a majority vote among all generated chains, we *vote over the top K complex chains*, as shown in Fig. 3.1C. In section 3.4.2, we will show that complexity-based consistency leads to further performance gains, on top of the existing gain from complexity-based prompting.

Putting everything together, our methods achieve new state of the art performance on three math benchmarks (GSM8K, MultiArith, and MathQA) and two BigBenchHard tasks (Date Understanding and Penguins) with substantial performance gains over Wei et al. [2022b]. We show that, compared with existing sample selection schemes, complexity-based prompting achieves better performance in most cases.

3.2 Related Work on In-Context Learning

Emergent Abilities and Multi-Step Reasoning With the recent trend in scaling language models [Brown et al., 2020a, Chowdhery et al., 2024], a central question is

what *unique* abilities emerge as models become large [Kaplan et al., 2020, Wei et al., 2022a]. Generally, the ability to follow the format of given prompts (typically few-shot) thus solving the corresponding tasks (also referred as in-context learning), is something that large language models are particularly skilled at [Shin et al., 2020, Liu et al., 2023]. Among the wide language understanding task spectrum, we are particularly interested in multi-step reasoning because of its two uniqueness: (1). multi-step reasoning is a task where large models substantially outperform smaller models [Wei et al., 2022b], versus performance gains on tasks like sentiment classification can be very limited with large models [Shin et al., 2020]; (2). multi-step reasoning is where few-shot prompting starts to outperform full training set fine-tuning, even when fine-tuning is conducted on the same large model [Lewkowycz et al., 2022]. This work takes an important step forward in multi-step reasoning by showing the critical role of prompt complexity.

Chain-of-Thoughts Reasoning A prominent work demonstrating the multi-step reasoning of language models is chain-of-thoughts prompting (Fig. 3.1A), proposed by Wei et al. [2022b]. They show that the reasoning ability can *only* be elicited by chain of thoughts, but not standard prompting where an answer directly follows a question without intermediate reasoning steps. Further works show that CoT can be improved by self-consistency [Wang et al., 2023b], pretraining the model with latex-formatted data [Lewkowycz et al., 2022], context selection [Creswell et al., 2023], or even adding certain magic phrases like “Let’s think step by step” [Kojima et al., 2022]. The original CoT paper [Wei et al., 2022b] uses 8 manually written examples as the prompt, which are reused by most follow-up works. Our work sits in the context of CoT reasoning, and propose a new complexity-based prompt selection that substantially outperforms the original CoT.

Example Selection for Prompting Designing prompts can be challenging due to the instability, as multiple works have shown the performance is sensitive to prompt, task, dataset, and model changes [Zhao et al., 2021, Lu et al., 2022, Su et al., 2022]. Despite works on automatic prompt searching (which is more suitable for smaller models, e.g., Shin et al., 2020, Li and Liang, 2021), currently, prompt engineering for large models is (still) a community-wide collective trial and error effort (there is even a prompt marketplace named PromptBase). The difficulty is that *it is extremely hard to extract generalizable regularity from empirical observations that can form effective selection criteria*. One notable exception is similarity-based prompt selection, which retrieves the most similar training instances as the prompt for a given test case [Rubin et al., 2022]. Yet for CoT prompting, retrieving different prompts for different test cases

requires reasoning chain annotations for the whole training set, which compromises the advantage of being few-shot. Given this background, our core contribution is identifying complexity as an effective and robust selection criterion and in many cases, it outperforms existing prompt selection schemes while being annotation-efficient.

Relation to Classical Semantic Parsing The procedure of chain of thoughts prompting is conceptually similar to classical semantic parsing where one generates a logical form then executes it upon a knowledge base to reach a final answer [Liang, 2016, Cheng et al., 2019]. The practice of sampling then voting is also similar to marginalizing out semantic parses [Yin et al., 2018]. There are further works linking the relationship between in-context learning and classical Bayesian inference [Wei et al., 2021a, Xie et al., 2022]. From our perspective, we tend to view chain-of-thoughts as flexible, language model styled “logical forms” which are “executed” by the language model itself. We leave further study on connecting classical parsing and CoT to future work.

3.3 Complexity-based Prompting

We study multi-step reasoning tasks, and use math word problems, mathematical problems expressed in natural language, as our testbed. This task, as is measured by solve rate (accuracy), is to predict the answer (typically a number) of a given math word problem via intermediate steps. We follow the chain-of-thoughts prompting framework and compare all prompting schemes using GPT-3 `text-davinci-002` and Codex `code-davinci-002`. An example problem, as well as the chain-of-thoughts workflow, is shown in Fig. 3.1A. The input is a stack of a few (often 8) CoT cases followed by a test question, then the language model continues generating an output CoT for the test question. Our goal is to improve the reasoning accuracy by identifying and exploiting more effective input and output reasoning chains.

3.3.1 Selecting Complex Samples as Prompts

Our method is to simply choose complex prompts over simple ones. We hypothesize that language models’ reasoning performance will increase if we use complex instances as in-context “training example,” as they intuitively subsume simpler instances [Richardson and Sabharwal, 2022]. We define complex instances as instances with more reasoning steps (Fig. 3.1B), as the name “multi-step reasoning” indicates. Note that us-

ing reasoning steps as the notion of complexity is also the practice of previous works like [Sugawara et al., 2018, Lai et al., 2021]. We further define a step as a line, separated by the linebreak “\n”.

There are two aspects that need more discussion: (1) The notion of complexity. There are other complexity indicators than number of steps, such as questions lengths or the length of the underlying formula for solving a given problem. We will show that the trend that better performance comes with more complex prompts is *consistent across various complexity indicators, such as question lengths and formula lengths*. Consequently, for datasets that do not have annotated reasoning chains, we can use questions lengths to identify complex instances, then only annotate the identified few-shot instances, thus reducing the annotation cost. (2) Confounders of number of steps. The increase in performance with more complex examples in the prompt could be explained by correlated factors like the increase in the total number of reasoning steps in the prompts or just the increased length of the prompt. To account for this, we evaluate prompts with simpler examples but the same number of reasoning steps (e.g. 24 cases with 3 steps vs. 8 cases with 9 steps, both of 72 steps in total). We also consider prompts of the longest lengths (but not most steps). We show that *the number of steps per example* is the most prominent source of performance gains over confounders.

3.3.2 Complexity-Based Consistency

Complexity-based prompting can be further enhanced with a new output selection method following the same intuition, which we present in this section. Existing evidence shows that the expressive neural models can take *shortcuts* during reasoning, relying on spurious correlations that inevitably exist in the training data [Mudrakarta et al., 2018, Sugawara et al., 2018, Lai et al., 2021]. This often leads to suboptimal generalization to unseen data. To alleviate this issue, we explicitly promote outputs with more complex reasoning chains at inference time. Specifically, our method follows the self-consistency practice in Wang et al. [2023b], which samples N reasoning chains for a test question. Different reasoning chains may lead to different answers, and Wang et al. [2023b] takes the majority answer as the prediction. In our case, instead of voting among all N chains, we only vote among top K ($K \leq N$) complex (more steps) reasoning chains, as shown in Fig. 3.1C. We dub our method *Complexity-based Consistency*. Note that when $K = N$ we recover the original self-consistency method. In our experiments, we set N to 50, and observe that the optimal K is always smaller than N (typically

30-40). This provides clear evidence that voting among more complex reasoning chains generalizes better than voting among all. We also show that if we do the opposite and vote among answers produced by K simplest reasoning chains, the accuracy is always worse than voting among all. This further validates that complex chains, not simple chains, should be considered more during decoding.

3.4 Experiments

We first discuss our experimental settings in section 3.4.1. In section 3.4.2 and 4.4.4, we present the following results: (1) our method substantially outperforms the original CoT [Wei et al., 2022b]. It establishes new state-of-the-art results on three math reasoning datasets (GSM8K; Cobbe et al., 2021; MultiArith; Roy and Roth, 2015; MathQA; Amini et al., 2019), a temporal reasoning task (Date Understanding; Suzgun et al., 2023), and the referential game task (Penguins; Suzgun et al., 2023). On StrategyQA [Geva et al., 2021], a commonsense reasoning dataset, our approach matches the existing state-of-the-art performance. (2) Performance gains from complex prompts are consistent: no matter what large model we use (GPT-3 or Codex), what distribution the prompt come from (in-distribution, noisy distribution, and distribution shift), or whether there exists prompt format perturbation or confounders, complex prompts consistently outperform simpler prompts; (3) Compared with other example selection schemes (random, heuristic and retrieval), complexity-based example selection often achieves the best or competitive results with minimal annotation budget. In the appendix, we discuss further experimental results, including a performance gain breakdown, results on smaller models, output step distribution, and further experiments on voting

3.4.1 Experimental Settings

Datasets We use three math word problems datasets (GSM8K, MultiArith, and MathQA) and three non-math reasoning (StrategyQA, Date Understanding, and Penguins) as our testbed. We choose GSM8K and MultiArith also because they are the datasets used by prior work on CoTs [Wei et al., 2022b, Wang et al., 2023b, Kojima et al., 2022], allowing fair comparison to existing methods. MathQA’s annotation are much noisier than others, and we use it to evaluate the robustness of our approach. There are 1.3K test instances in GSM8K, 600 in MultiArith, and 600 in MathQA. For

each dataset, we randomly draw 200 instances from the training data to create a validation split. The cost of prompting GPT-3 is proportional to the size of test set. For the non-math datasets, StrategyQA is a multi-step commonsense reasoning task with 800 test instances. Date Understanding is a temporal reasoning task with 250 test instances. Penguins is a referential game (a referential game asks questions referring to different objects, e.g., is penguin A older than penguin B and C) with 146 test instances. Both Date Understanding and Penguins are subsets of the BigBench Hard datasets (datasets that previously fine-tuning struggles with, see Suzgun et al., 2023). Evaluating on a 200-instances validation set costs about 6-8 US dollars for greedy decoding (1 output chain) and \$12-\$24 for sampling 50 output chains. Prompting Codex is currently (November 2022) free and we hope OpenAI could continue making it free to the community.

Language Models We consider two paradigms: fine-tuning and prompting. For fine-tuning, we report the existing SOTA performance: a fine-tuned GPT3 with a verifier [Cobbe et al., 2021] on GSM8K, a relevance and LCA operation classifier [Roy and Roth, 2015] on MultiArith and a customized sequence to sequence model [Amini et al., 2019] on MathQA. For prompting, we consider the following language models: (1). LaMDA [Thoppilan et al., 2022], a 137B model used as the baseline in Wei et al. [2022b]; (2). PaLM [Chowdhery et al., 2024], the primary 540B model used in the CoT papers; (3). Minerva [Lewkowycz et al., 2022], a 540B large model that trains on \LaTeX data; it achieves SOTA performance in math reasoning on GSM8K; (4). GPT-3 175B (text-davinci-002 from Brown et al., 2020a) (5). Codex (code-davinci-002 from Chen et al., 2021b, also 175B). We further consider the DiVeRSe [Li et al., 2023c] method which equips an additional trained verified to GPT-3/ Codex and is the previous SOTA on GSM8K. Our experiments are mostly conducted on GPT-3 and Codex because they are the accessible to the public thus more reproducible. LaMDA, PaLM and Minerva are not accessible to the public, and their numbers are from their corresponding papers.

Prompts and Hyperparameters The training sets of GSM8K and MathQA contain human annotated reasoning chains, within which we search for complex prompts. MultiArith does not have annotated reasoning chains, so we consider two strategies. (1). *in-distribution annotation*, which uses question lengths as an alternative proxy for complexity, then manually annotates reasoning chains for complex questions; (2). *prompts transfer* from GSM8K training data. All prompts for math datasets contain 8 cases (a case = a question + a chain of thoughts + an answer). For non-math datasets, since they

Table 3.1: Complexity-based prompting, when applied on Codex (code-davinci-002), achieves new state-of-the-art performance on GSM8K, MultiArith, and MathQA. † models are not publicly accessible, and the numbers are from their papers. Our performance gain (**+blue**) is computed over the original handcrafted CoT used in Wei et al. [2022b], which is our primary baseline. Our methods substantially increase the performance over Wei et al. [2022b], with an average **+5.3** gain on GPT-3 and **+6.2** on Codex.

		GSM8K	MultiArith	MathQA
Previous finetuning SOTA		57.0	60.5	37.4
Greedy decoding [Wei et al., 2022b]				
LaMDA† [Thoppilan et al., 2022]		17.1	51.8	-
PaLM† [Chowdhery et al., 2024]		58.1	94.7	-
Minerva† [Lewkowycz et al., 2022]		58.8	-	-
Text-davinci-002	Handcrafted CoT	48.1	90.8	30.1
	Random CoT	49.7	89.5	34.8
	Complex CoT	55.4 (+7.3)	94.2 (+3.4)	36.0 (+5.9)
Code-davinci-002	Handcrafted CoT	61.0	95.8	29.3
	Random CoT	60.4	97.3	40.5
	Complex	66.6 (+5.6)	95.8 (+0.0)	47.3 (+18.0)
Voting among multiple outputs [Wang et al., 2023b]				
LaMDA† [Thoppilan et al., 2022]		27.7	75.7	-
DiVeRSe [Li et al., 2023c]		82.3	99.8	-
PaLM† [Chowdhery et al., 2024]		74.4	99.3	-
Minerva† [Lewkowycz et al., 2022]		78.5	-	-
Text-davinci-002	Handcrafted CoT	64.0	98.2	43.8
	Random CoT	62.0	95.2	48.5
	Complex CoT	71.5	97.3	49.5
	+ Vote Complex	72.6 (+8.6)	98.7 (+0.5)	50.2 (+6.4)
Code-davinci-002	Handcrafted CoT	74.6	99.7	55.0
	Random CoT	77.3	99.3	58.2
	Complex CoT	82.6	99.7	58.6
	+ Vote Complex	82.9 (+8.3)	99.8 (+0.1)	60.0(+5.0)

do not have annotated reasoning chain, we again, use question length as the complexity proxy and manually annotates reasoning chains for complex questions. Following Kojima et al. [2022], we add “Let’s think step by step” before the reasoning chains for all prompting schemes to improve the performance.

3.4.2 Main Results

Overall Test Performance on Math Datasets Table 4.1 shows the overall performance of models. We consider two decoding strategies: (1) greedy decoding (the first block of Table 4.1) and (2) majority vote (ssec:method:consistency; the second block of Table 4.1). Note that PaLM and Minerva are more than three times larger than GPT-3 and Codex, the model we use to evaluate our method, and Minerva is additionally pretrained on latex data. Therefore, they are by no means comparable to the methods based on GPT-3 or Codex. We nevertheless outperform all of them.

We consider three prompting schemes: (1). *Handcrafted CoT* constructed originally by Wei et al. [2022b] then reused in following-up works [Wang et al., 2023b, Kojima et al., 2022, Wang et al., 2022a]. (2). *Random CoT*: randomly drawing samples from the training set. GSM8K and MathQA training data have reasoning chain annotations, so we directly use them. MultiArith does not have reasoning annotations, so we randomly sample eight training cases then annotate the chains manually. (3). *Complex CoT*. For GSM8K and MathQA, we choose eight training cases with the most numbers of reasoning steps; For MultiArith, we use the question length as the proxy for complexity, and manually annotate reasoning chains for the eight training cases with the longest questions. Complex prompt selection results in substantially more reasoning steps: it averages 9.0 steps on GSM8K, while the handcrafted and random schemes yield 3.4 and 2.8 steps respectively. The trends are similar on the other two datasets. The handcrafted prompts uses the same fixed prompt for all three datasets but the cases within the prompt does not come from any of the datasets (so they are in a sense, out of distribution). Complex prompts and random prompts all come from their corresponding training sets (so these two are in a sense, in-distribution).

As Table 4.1 shows, our method achieves substantially better performance than the baselines. Besides, our proposal of voting among complex chains outperforms voting among all (last two lines in Table 4.1. Furthermore, our performance using GPT-3 is close to PaLM and Minerva, two language models that are more than three times larger than GPT-3 and are not publicly accessible. These results directly demonstrate

Table 3.2: Complex prompts give comparable performance to PaLM on StrategyQA (commonsense reasoning), and achieve new state of the art performance on Date Understanding (temporal reasoning), and Penguins (referential game) datasets. Accuracy gain (**+blue**) is computed over the original handcrafted CoT used in Wei et al. [2022b,a]. All results use greedy decoding.

	Prompt	Strategy	Date	Penguins
PaLM	Handcrafted	77.8	79.2	65.1
Text-davinci-002	Handcrafted	66.9	82.8	76.7
	Simple	71.1	76.4	61.0
	Complex	77.0 (+10.1)	82.4 (-0.4)	79.5 (+2.8)
Code-davinci-002	Handcrafted	73.1	86.0	78.1
	Simple	74.4	83.2	69.8
	Complex	73.9 (+0.8)	86.8 (+3.6)	80.8 (+2.7)

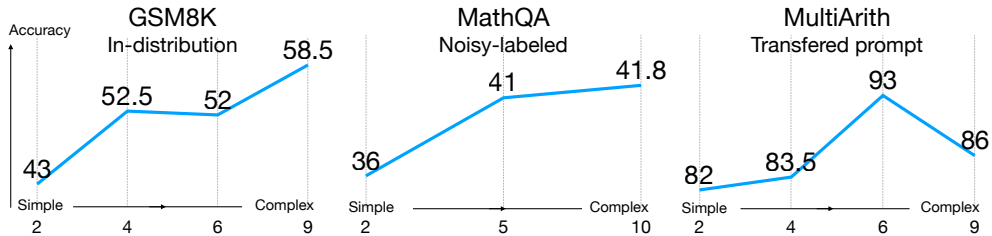


Figure 3.2: Validation set performance. X-axis means reasoning steps and y-axis means accuracy. More reasoning steps in prompts overall achieve higher accuracy when prompts are in-distribution (left), noisily labeled (middle), and out of distribution (right).

the effectiveness of our methods.

Consistent Performance Improvements on Different Reasoning Tasks Table 3.2 shows that the advantage of complex prompts holds for different types of reasoning tasks. When prompted with complex examples, GPT-3/ Codex achieves new SOTA performance on Date Understanding and Penguins datasets where complex prompts consistently improves performance over simpler prompts.

Consistent Performance Improvements in Different Prompt Distributions We investigate the performance of our complexity-based prompting when the prompts are: (1) from clean in-distribution training set (GSM8K); (2) from noisy annotation (MathQA);

Table 3.3: Comparison to other prompt example selection schemes (validation accuracy). On GSM8K and MultiArith, complexity-based selection outperforms all the baselines. On MathQA, although retrieval performs better than complexity, it requires substantially more annotation.

	#Annotations	GSM8K	MultiArith	MathQA
Random	Few-shot (8)	52.5	86.5	33.0
Centroid	Few-shot (8)	52.0	92.0	32.0
Retrieval	Full training set (≥ 10000)	56.0	88.0	69.5
Complexity (ours)	Few-shot (8)	58.5	93.0	42.5

(3) are transferred from another dataset (MultiArith). Here as MultiArith does not have annotated reasoning chains, and their questions are similar to the ones in GSM8K; we use (transfer) prompts from GSM8K for MultiArith. Figure 3.2 shows that in general, more complex prompts achieve better performance, and this trend is consistent in all the three settings, except for one particular case on MultiArith.

Comparison to other Example Selection Schemes As we view the reasoning complexity as the basis of a new example selection scheme, we compare it with existing selection schemes. We consider: (1) *random* selection; (2) *Centroid*, where we select examples whose question embeddings (produced by a pretrained sentence encoder Reimers and Gurevych, 2019) are the closest to the embeddings of all other questions, i.e., questions at the center part of the dataset. The intuition is that centroid examples may be the most typical or representative cases of a dataset; (3) *Retrieval*, where we retrieve questions from a training set whose embeddings are closest test question measured in Euclidean distance. Notably, there are important differences between retrieval and other methods: retrieval uses different prompts for different test cases, while other methods use fixed prompts for all. Therefore, the annotation cost of retrieval scales with the size of the test set, and is usually about the full-training-set-sized annotation (more than 10K cases), while others only require few-shot annotation (in our cases, only 8 examples).

As shown in Table 3.3, complexity-based selection outperforms all other methods on GSM8K and MultiArith. On MathQA, although retrieval-based selection outperforms complexity-based selection, it has two importance restrictions that we do not have: (1) as mentioned, retrieval requires substantially more CoT annotation, while we only requires few-shot; (2) the performance of retrieval is critically determined by how

Figure 3.3: X-axis means reasoning steps of dev set cases and y-axis frequency. The direction of generalization on the two datasets is intriguing and show different patterns: on GSM8K, simple prompts perform better for simple cases (≤ 3 steps) while complex prompts perform better for complex cases; on MathQA, simple prompts do not have advantages for simple case and complex prompts seem to perform better on most of the groups.

similar the test cases and the training questions are to each other, and the similarity may not always hold. We further note that on MathQA, many dev. questions are quite similar to their retrieved training questions (some of them only have minor changes like “8 apples plus 9 bananas” to “10 apples plus 5 bananas” while the underlying computations are the same). So in general, complexity-based prompting has the advantage of good performance while being annotation efficient.

Direction of Generalization Intuitively, one may attribute the improvements of complexity-based prompting to accuracy gains on complex test cases. Yet interestingly, our analysis suggests the opposite. Fig. 3.3 compares the validation set accuracy of complex and simple prompts, varying the number of reasoning steps in the *gold annotation*. We observe a clear trend on both GSM8K and MathQA: complex prompts perform on par with simple prompts on *hard* cases, while achieving more clear gains on cases with fewer number of reasoning steps. This finding suggests that complexity-based prompting generalizes to simpler test cases. We conjecture that this is because the reasoning capabilities elicited by complex prompts may cover simple questions better. Further investigation into the underlying mechanism is definitely interesting, and is left to future work.

3.4.3 Analysis

In this section, we develop in-depth analysis supporting our claims. All experiments in this section are performed on validation sets. We first show that the performance improvements with more reasoning complexity is consistent in terms of: (1). different proxies for complexity and (2). different step formatting. Then we show that the number of reasoning step is the most prominent factor for performance improvements over its confounders. Finally, we strengthen our conclusion of complexity-based consistency, and show that the optimal performance is always achieved by majority voting over complex chains, not simple chains.

Table 3.4: Alternative complexity measure: Q Len. = question length, F Len. = formula length. More complex prompts consistently outperform simpler ones.

	Q Len.	GSM8K	F Len.	MathQA
Simple	70	49.0	7.5	37.5
Mid	226	51.0	55	33.5
Complex	815	52.5	165	43.5

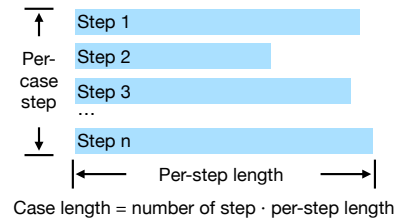


Figure 3.4: Relationship between confounders.

Table 3.5: Sensitivity analysis on step formatting. Complex prompts consistently lead to better performance with regard to different step formatting.

	Linebreak “\n”	Period “.”	Explicit “step i”	Semicolon “;”
GSM8K-Complex	58.5	54.5	52.0	54.0
GSM8K-Simple	43.0	40.5	42.0	41.0
MathQA-Complex	42.5	39.0	36.0	39.5
MathQA-Simple	34.0	34.5	33.5	37.0

Alternative Proxies for Complexity Complexity-based prompting is equally applicable when the data does not come with reasoning chain annotations, as we have already shown that selecting cases with longest questions also improves performance (§3.4.2). In Table 3.4, we confirm that in addition to number of steps, either using questions length or formula length as the measure of complexity, the optimal performance is achieved with complex prompts. These results mean that the effectiveness of complex prompts are consistent with regard to the notion of complexity.

Sensitivity Analysis on Step Format A common concern with prompting is that the performance can be sensitive to the format of the input [Shin et al., 2020, Liu et al., 2022a] and may change with input perturbations. Here we study one important perturbation: the splitter of steps, which is an existing concern of CoT-styled prompting in Rong, Akyurek and Akyurek [2022]. As alternatives to the linebreak “\n” we use, we consider two more types of splitters: (1). explicit phrases “step i” (2). two punctuation marks, period “.” and semicolon “;” The performance is shown in Table 3.5. Although these perturbations do have an influence on the performance, complex prompts consistently lead to better performance with regard to different step formatting.

Table 3.6: Confounder analysis. Although there exist confounders like number of cases or total prompt length, the number of reasoning step is the most prominent factor for performance gain given moderate per-step length.

<i>More number of simple cases v.s. less but complex cases</i>	GSM8K		MathQA	
Total reasoning step	72	72	45	45
Number of cases in prompt	24	8	19	8
Per-case reasoning step	3	9	2.25	5.625
Accuracy	51	58.5	37.5	42.5
<i>Most number of reasoning steps v.s. longest prompt</i>	GSM8K		MathQA	
Number of cases in prompt	8	8	8	8
Length of prompt	12.6k	8.4k	7.6k	4.9k
Number of total reasoning step	59	72	32	45
Per-step length	112	74	137	52
Accuracy	57	58.5	31	42.5
<i>Shorter per-step length v.s. Longer per-step length</i>	GSM8K		MathQA	
Number of total reasoning step	72	72	45	45
Number of cases in prompt	8	8	8	8
Per-step length	36	74	37	52
Accuracy	51.0	58.5	30.5	42.5

Confounder Analysis All experiments so far keeps the number of instance to be 8 in all prompts. Yet when choosing complex examples with more reasoning steps, we observe that the following factors are correlated (also illustrated in Fig. 3.4): (1). when *per-case reasoning step* increases (for example, in GSM8K we choose cases with 9 reasoning steps), the *total number of step* in the whole prompt also increase (in GSM8K, we have 8 cases in the prompt, so there are $8 \times 9 = 72$ steps in total). This might be compensated by using *more number of simple cases* (e.g., 24 simple cases, each with 3 steps, can also make 72 steps in total). These factors are shown in the upper part of Table 3.6. (2). when *per-case step* increases, the *full length of the prompt* (= number of characters) also increases, which may be compensated by longer (more characters) but less step examples. These factors are shown in the lower part of Table 3.6. From the accuracy results we can see that: (1). keeping full number of reasoning steps the

Table 3.7: Progress about prompt optimization on MATH after our complexity-based prompting work. This field has been advanced significantly in recent time and the most SOTA method nearly saturates the MATH benchmark (which is so far the most challenging STEM reasoning benchmark). We are excited to see how this field will continue to improve.

Model	Prompting Method	Release time	Acc
GPT-4	Default Prompt	Mar 2023	42.5
GPT-4	Complexity-based Prompting [Zheng et al., 2023]	Aug 2023	50.3
GPT-4	PHP Prompting [Zheng et al., 2023]	Aug 2023	53.9
GPT-4	Code-based Verification [Zhou et al., 2024]	Aug 2023	73.5
Claude-3	Default Prompt [Anthropic, 2024]	Mar 2024	60.1
LLaMA-3	Default Prompt	Apr 2024	50.4
GPT-4o	Default Prompt	May 2024	76.7
Gemini 1.5 Pro Math	Default Prompt [Reid et al., 2024]	May 2024	80.6

same, using more number of simple cases does not outperform less number of complex cases; (2). longest prompts does not outperform complex prompts. (3). yet we do need a moderate per-step length because keeping total number of step 72, moderate per-step length prompts outperforms shorter per-step length prompts. This means that despite the existence of confounders, the number of reasoning steps per example is the most prominent factor for performance gain given moderate per-step length.

3.5 Summary and Follow-on Work

This chapter proposes a new complexity-based instance selection scheme for prompting language models to perform multi-step reasoning. In addition to substantial performance improvements on math word reasoning tasks, our methods exhibit multiple advantages such as being intuitive, annotation-efficient, and robustly effective in different in-context learning settings.

After the initial paper of this work released in October 2022, there has been significant progress in the field of reasoning, which we list on Table 3.7 using the MATH benchmark [Hendrycks et al.], a more challenging dataset than the GSM8K dataset we use. We see that our complexity-based prompting, when applied on the MATH dataset, improves the baseline from 42.5 to 50.3. Afterward, this area remains ac-

tive, and notable methods that clearly improve the performance include: (1) stepwise supervision and feedback [Lightman et al., 2023, Wang et al., 2023a]; (2) code-based verification [Zhou et al., 2024]. The most recent work from Gemini 1.5 Pro [Reid et al., 2024] improves the accuracy to 91.1, nearly saturating this benchmark. We tend to believe that future reasoning research may require evaluation on complex, long-horizon, real-world tasks such as SWE-Bench [Jimenez et al.], and are excited to see how this area will continue to improve.

Chapter 4

Improving Reasoning by Instruction Finetuning

Having discussed in-context learning, in this chapter, we delve deeper into model finetuning. The surprising ability of Large Language Models (LLMs) to perform well on complex reasoning with only few-shot chain-of-thought prompts is believed to emerge only in very large-scale models. We show that such abilities can, in fact, be distilled down from GPT-3.5 to T5 variants. We propose *model specialization*, to specialize the model's ability towards a target task. The hypothesis is that large models (commonly viewed as larger than 100B) have strong modeling power such that they can perform a large spectrum of tasks. Small models (commonly viewed as smaller than 10B) have limited model capacity, but if we specialize their capacity towards a target task, the model can achieve decent performance improvements. We use multi-step math reasoning as our testbed because it is a very typical emergent ability. We show two important aspects of model abilities: (1) balancing the language model's performance on multiple tasks is a delicate matter, as improvements on one task may compromise other tasks; (2) yet by intentionally paying the price of decreased generic ability, we can clearly improve across different model scales smaller than 10B towards a specialized multi-step math reasoning ability. We further give comprehensive discussions about important design choices for better generalization, including the data format mixture and the start model checkpoint.

4.1 Introduction

Recently, the field of NLP is significantly impressed by large language models’ strong abilities [Brown et al., 2020b, Chowdhery et al., 2024]. Wei et al. [2022a] discuss the emergent abilities of large language models – abilities that only exist in large models (more than 100B parameters), but not in smaller ones. A typical example (also the first-discovered one) is the ability to perform multi-step reasoning on math word problems with chain-of-thought (CoT) prompting [Wei et al., 2022b] where the authors let the model generate a step-by-step reasoning chain to help get the final answer. The existence of such abilities has a profound impact: on the positive side, such abilities open countless opportunities for new research directions; on the negative side, very few organizations have the compute to even fine-tune 100B-scale models, making such abilities less accessible. It would be ideal if smaller models can also obtain emergent abilities like math CoT reasoning, so they can be accessed by a larger range of researchers and practitioners. However, preliminary results of Wei et al. [2022a] show that if the model scale is small (empirically less than 100B parameters), CoT exhibits flat, sometimes even near zero scaling curve [Wei et al., 2022b]: the model CoT performance benefits little from scaling up, until a certain threshold is reached. Later smaller models’ scaling curve is partially improved in Chung et al. [2024], but they still underperform large models. These results so far are rather pessimistic since they suggest improving CoT performance for smaller models can be challenging. At the current stage, the community is eager to know to what extent such abilities can be further improved in smaller models.

This paper addresses the problem of CoT reasoning for smaller models by *model specialization*. Our hypothesis is that large models ($\geq 100\text{B}$) have strong modeling power but are spread over a large spectrum of tasks. Small models ($\leq 10\text{B}$) have limited model capacity, but if we “concentrate” their capacity on a target task (improving one target task at the cost of decreased capability on other tasks), the model can achieve decent performance improvements. There exists promising preliminary work on smaller models’ chain-of-thought abilities such as UL2 [Tay et al., 2023] and FlanT5 [Chung et al., 2024], but their approaches risk spreading the models’ limited capacity across many different datasets, hurting their performance in tasks involving complex reasoning. In our experiments, we show that we can improve the scaling trend of CoT reasoning on small FlanT5 models (250M, 760M, and 3B) by a large margin (an average +10 accuracy gain) on a suite of 4 math reasoning tasks (1 in-distribution and 3 out-of-

distribution). This might come at a cost of decreased performance in other benchmark tasks (specifically in this paper, worse performance on the BigBench Hard suite Suzgun et al., 2023). This means that we can indeed move the model’s power from generic abilities to concentrate on the target math CoT.

Our approach is to fine-tune an instruction-tuned model (FlanT5) by distilling chain-of-thought reasoning paths of the GSM8K data from a large teacher model (GPT-3.5 code-davinci-002 Chen et al., 2021b), then do a model selection on the average performance of three held-out math reasoning data to ensure the model’s out-of-distribution generalization. Although distillation per se is a well-studied area, there are multiple caveats in our process, as we will demonstrate: (1) the teacher model (code-davinci-002) and our student model FlanT5 use different tokenizers, we address the tokenizer alignment problem by dynamic programming. (2) Distillation induces different performance on an instruction-tuned checkpoint (in our case, FlanT5) and the raw pretrained checkpoint (T5), where specialized FlanT5 performs better but specialized T5 achieves more accuracy gain. (3) at the late training stage, the model’s in-distribution and out-of-distribution (OOD) performance fluctuates differently, so if one wants better OOD generalization, the model selection should be performed on held-out math datasets, rather than the validation portion of the tuning data. (4) multiple tradeoffs happen during the distillation/ specialization process: as we start distillation, on BigBench Hard test suite (the measure of generic ability), the model immediately loses all its CoT prompting abilities, and gradually loses a large portion (but not all) of answer-only prompting abilities. The data format we use for tuning is also closely related to model ability: in-context examples enable both in-context and zero-shot performance, but zero-shot examples lose the model’s in-context ability for increased zero-shot ability.

These findings deepen our understanding of language model chain-of-thought reasoning behavior in multiple aspects: (1) the previous hypothesis is that CoT has near-flat scaling curves on a small scale, we show that we can lift up the scaling curve (i.e., improve model performance across scale) by “concentrating” the model’s capacity on a target ability. This shows that although for pretrained checkpoints, CoT might be a capability that only exists in large models, after specialization, smaller models’ scaling curves become log-linear, just like large models [Kaplan et al., 2020, Hoffmann et al., 2022]. (2) previous observation of LLM behaviors indicates complex tradeoffs and balances of model ability across multiple tasks, we give a detailed description of how we move the model’s power from generic abilities to a target ability, clearly showing what can be gained at what cost. (3) common practice selects the model on the validation

split of the same dataset, we select the model based on the performance of different math reasoning datasets, to alleviate overfitting on a single dataset.

4.2 Background

Specialized Language Models Although modern language models show strong generic abilities on multiple directions, recent analysis [Fu et al., 2022a] shows models do have different focuses (e.g., code-davinci-002 for code and text-davinci-003 for text). Ability tradeoff happens at all scales: for large models, such a tradeoff does not have to be all or nothing: code-davinci-002, although specialized for code, can still solve a lot of text problems; for small models, due to limited model capacity, they have to trade all generic abilities for one special ability. One example is GitHub Copilot, which supposedly is a 12B small model [Thakkar, 2022]. The actual practice of specialization is simply finetuning: to specialize a model towards a target ability, one simply tunes the model using the related data, which is the practice of concurrent work about smaller models' CoT ability [Magister et al., 2023, Shridhar et al., 2023, Ho et al., 2023]. The problem here is how to generalize beyond the tuning data, as small models may simply overfit the tuning distribution but struggle to generalize when the distribution shifts [Liu et al., 2022b, Si et al., 2022]. So far the community's hypothesis of OOD generation involves two important aspects: (1). model scale [Chowdhery et al., 2024]; (2). instruction tuning [Chung et al., 2024], which we will also study. These factors mark the differences between our work and the concurrent distillation work: we show how the model trades generic abilities for the target ability, and how model scale and instruction tuning help the model gain better in-distribution and OOD performance.

Distillation and Data Augmentation Our approach of using data generated from code-davinci-002 to tune smaller FlanT5 can be viewed as either distillation [Tan et al., 2019] or data augmentation [Li et al., 2024]. Here we note that we merely use the generated data as the tool for model specialization, and the specialization data can also be from other sources like human annotation. Our focus is to study the ability tradeoff during specialization, but not directly contribute to the distillation or data augmentation literature.

Most closely related works There are two threads of most related works: (1). FlanT5 [Chung et al., 2024] and UL2 [Tay et al., 2023] which is the first work discussing smaller models' CoT ability, but they focus on generic CoT while we trade generic ability for math CoT. (2). language model self-improvement [Huang et al., 2023] which

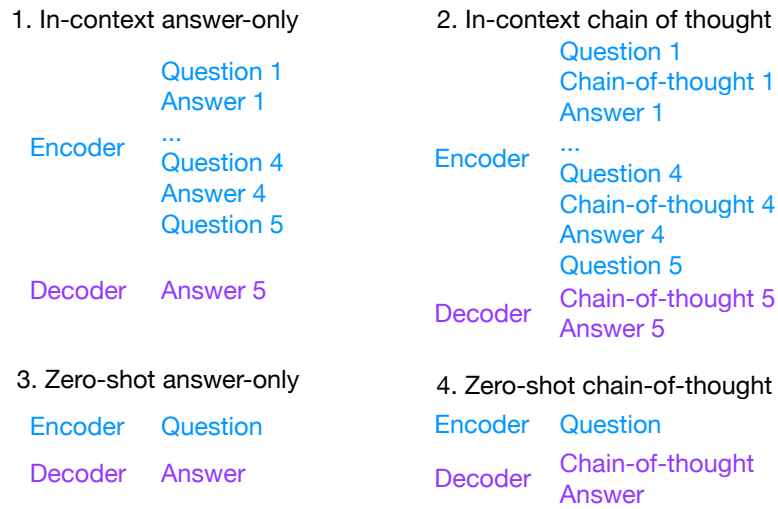


Figure 4.1: Four data formats we consider for tuning the model. We will show tuning with in-context chain-of-thought examples is particularly important for the model’s CoT ability.

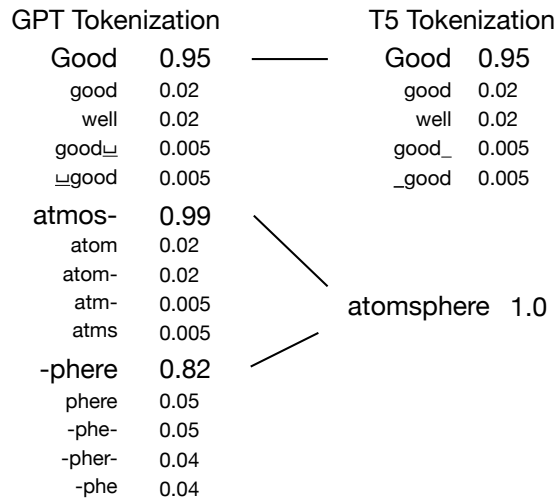


Figure 4.2: Aligning GPT tokenization to T5 tokenization by dynamic programming. If a T5 token has a one-to-one alignment to a GPT token, we reuse the GPT’s top 5 probability as the target distribution. If there the mapping is one-to-many/ many-to-one, we treat the T5 token’s distribution as one-hot.

also use CoT data augmentation, but they only consider large models and do not show the tradeoff between model abilities. Here we focus on small models and clearly show the price for ability improvements.

4.3 Specializing Multi-Step Reasoning

Our objective is to study what it takes to improve smaller models’ chain-of-thought math reasoning. We use GSM8K [Cobbe et al., 2021] as our seed dataset because it is one of the datasets with most diverse math reasoning problems, but test the model’s performance of three additional math datasets (MultiArith, ASDiv, and SVAMP Wei et al., 2022b) to show the model generalizes to OOD data. We further use BigBench Hard to test to model’s generic reasoning ability, demonstrating the tradeoff between generic and target abilities. We use T5 (raw pretrained checkpoint) and FlanT5 (instruction tuned checkpoint) as our base model, and use code-davinci-002 to generate distillation/ specialization data.

Distillation from Code-Davinci-002 Given a training question corpora, we use code-davinci-002 to generate 40 new CoT solutions then take the ones that lead to the correct answers as our training data. One solution consists of an answer and a chain of thought explaining the intermediate steps towards the answer. In addition to the standard finetuning setting where one uses the question as the input and use the [CoT, answer] pair as the output (Fig. 4.1), we further consider three additional data formats: (1). in-context answer-only, where we do not use the CoT data (hence the name “answer-only”) and prepend 4 in-context examples before the question (hence the name “in-context”). The reason we prepend the in-context example is that previous work shows tuning with in-context examples improves the model’s in-context learning ability [Min et al., 2022]. (2). in-context chain-of-thought, where we add CoT to both the in-context example and the output. (3). zero-shot answer-only, where we directly input the question and output the answer. Using answer-only data is because previous work shows they improve performance. In our experiments, we will show that in-context data induces zero-shot ability but zero-shot data sacrifice in-context learning ability. We note that there also exist techniques like adding a calculator [Cobbe et al., 2021] or self-consistency decoding [Wang et al., 2023b] that can further improve the performance. These techniques are orthogonal to the distillation we use and can definitely be integrated to our work for better performance. Since our focus is the balance of the models’ special and generic abilities, we leave the integration of these orthogonal techniques to future work.

In terms of training objectives, in the distillation literature, there are typically two types of distillation approaches: (1). sample matching, where one trains the student model on the data generated by the teacher. In our case, sample matching means we

directly optimize the student’s likelihood on the data generated by code-davinci-002. (2). distribution matching, where one minimizes the KL divergence between the student’s output distribution (in our case, the per-step autoregressive distribution) and the teacher’s. Usually, distribution matching is shown to achieve faster convergence and better performance than sample matching, so we use distribution matching as our training objective. However, the OpenAI API only grants access to the 5 most probable tokens at each decoding step, but not the probability distribution over the entire vocabulary. This is not necessarily a bad thing: although the per-step distribution only covers the top 5 tokens, most of the time their probability sum is close to 1, being a good enough approximation of the full vocabulary distribution. We set to zero the probabilities of tokens not in the top 5. Note that even if one could access the full-vocabulary distribution, storing them on disk will be nontrivial as the full-vocabulary distribution takes about 10k times larger space.

Aligning tokenizers by dynamic programming One problem when matching the two distributions is the misalignment between the GPT tokenizer and the T5 tokenizer. We solve this problem by dynamic programming. Specifically, given two sequences of tokens ($\mathbf{s}_{1:L}, \mathbf{t}_{1:N}$), our objective is to find an alignment that minimizes the total cost of editing one sequence to the other. Our dynamic program is a slight tweak of the textbook dynamic programming algorithms used in bioinformatics for sequence alignment (such as the Needleman–Wunsch algorithm Needleman and Wunsch [1970]) and in signal processing (such as dynamic time wrapping Senin [2008]). The recursion function is:

$$f(i, j) = \min\{f(i-1, j) + c(\mathbf{s}_i, \mathbf{t}_j), \quad (4.1)$$

$$f(i, j-1) + c(\mathbf{s}_i, \mathbf{t}_j), \quad (4.2)$$

$$f(i-1, j-1) + c(\mathbf{s}_i, \mathbf{t}_j)\} \quad (4.3)$$

where $f(i, j)$ denotes the total cost aligning $\mathbf{s}_{1:i}$ and $\mathbf{t}_{1:j}$ and $c(\mathbf{s}_i, \mathbf{t}_j)$ is the predefined string edit distance between token \mathbf{s}_i and \mathbf{t}_j . Our algorithm does not enforce one-on-one matching between tokens in the two sequences, and one token in \mathbf{s} might align with multiple in \mathbf{t} and vice versa. Fig. 4.2 gives an example alignment. If there exists a one-to-one mapping between a GPT token and a T5 token, we use the GPT distribution as the T5 distribution. If the mapping is not one-to-one, e.g., two T5 tokens map to one GPT token, or two GPT tokens map to one T5 token (Fig. 4.2 lower part), we do not use the corresponding GPT distribution and set the T5 distribution to be one-hot. We further note that aligning sequences generated by different tokenizers is a generic problem of

contemporary NLP, yet we are not aware of any existing libraries approaching it. In our open-sourced code, we also release the implementation of our dynamic program and hope it can be useful for future research.

4.4 Experiments

The objective of the experiments is to see to what extent we can push up the scaling curve of smaller models' math CoT performance and what is the price of it. We conduct model specialization on two model families: the raw pretrained checkpoints, and their instruction-tuned checkpoints (recall that the instruction-tuned checkpoints are generally more capable than the raw pretrained checkpoints. Specifically, we consider the raw pretrained T5 Base (250M)/ Large (760M)/ XL (3B)/ XXL (11B), and the instruction-tuned FlanT5s. In Sec. 4.4.1, we validate our main hypothesis that large models can perform well on a wide range of tasks while smaller model's ability can be moved from generic abilities to a specialized target ability. Specifically, we show model specialization can indeed improve CoT math performance for FlanT5-Base/ Large/ XL/ XXL, while paying the price of generic abilities, i.e., losing all CoT abilities on Big-Bench Hard and a large portion of answer-only (AO) abilities. In Sec. 4.4.2, we study the scaling behavior of smaller models and show how specialization lifts up the scaling curve for both T5 and FlanT5. This modifies the previous belief that smaller models exhibit a flat scaling curve [Wei et al., 2022b]; we show that their scaling curve becomes log-linear after specialization, but not flat. In Sec 4.4.3, we show the dynamics and the generalization behavior of specialization: the model's target performance increases gradually but generic abilities decrease gradually during tuning, and there exists tradeoffs between in-distribution v.s. OOD performance and in-context v.s. zero-shot performance.

4.4.1 Overall Performance Tradeoff

We test the models' math reasoning ability and generic ability and show their tradeoffs. For the math reasoning ability, we use the code-davinci-002 augmented GSM8K dataset [Cobbe et al., 2021] as our tuning dataset. The GSM8K has 7K training questions, for each question we ask the large model to generate 40 different solutions, taking the correct ones from the generation, we have 130K tuning data points in total. We test the model's out-of-distribution performance on MultiArith, ASDiv, and SVAMP (col-

Table 4.1: Overall test set performance. We specialize Flan-T5’s ability from the generic tasks (BigBench Hard) to math reasoning tasks. After paying the cost of BigBench Hard performance (the model loses all the CoT prompting ability and a large portion of the Answer-only (AO) prompting ability), we see the specialized T5 models have improved in-distribution (GSM8K) performance (where our 3B and 11B models outperform concurrent works) as well as out-of-distribution (MultiArith, ASDiv and SVAMP) performance, showing that we can move the model’s ability from generic tasks (BBH) to a specific target task (math reasoning). Magister22: Magister et al. [2023]; Shridhar22: Shridhar et al. [2023]; Ho22: Ho et al. [2023].

Models	CoT Reasoning on Maths Word Problems								BigBench-Hard			
	GSM8K		MultiArith		ASDiv		SVAMP		AO		CoT	
	Acc.	Δ	Acc.	Δ	Acc.	Δ	Acc.	Δ	Acc.	Δ	Acc.	Δ
code-davinci-002	63.1	-	95.8	-	80.4	-	76.4	-	56.6	-	73.9	-
LaMDA	14.8	-	45.0	-	46.6	-	37.5	-	-	-	-	-
PaLM	29.9	-	75.0	-	61.9	-	46.7	-	37.4	-	43.0	-
UL2	4.4	-	-	-	16.9	-	12.5	-	-	-	-	-
Concurrent Works with Knowledge Distillation												
Magister22, T5	21.9	-	-	-	42.1	-	-	-	?	-	?	-
Shridhar22, GPT	21.0	-	-	-	-	-	-	-	?	-	?	-
Ho22, GPT	6.8	-	33.3	-	-	-	-	-	?	-	?	-
Our Specialized Models Compared with Baselines												
FlanT5-XXL	16.1	-	51.7	-	36.5	-	39.7	-	47.4	-	41.8	-
+ Specialize	27.1	+11.0	63.0	+11.3	37.6	+1.1	35.6	-4.1	19.6	-27.8	0.0	-41.8
FlanT5-XL	13.5	-	24.0	-	20.7	-	17.7	-	39.9	-	35.8	-
+ Specialize	22.4	+8.9	42.3	+18.3	28.4	+7.7	23.8	+6.1	3.2	-36.7	0.0	-35.8
FlanT5-Large	6.9	-	13.0	-	10.1	-	6.8	-	30.3	-	30.9	-
+ Specialize	20.2	+13.3	38.5	+25.5	23.8	+13.7	20.4	+13.6	6.5	-23.8	0.3	-30.6
FlanT5-Base	3.0	-	7.0	-	4.2	-	3.8	-	24.2	-	25.9	-
+ Specialize	13.4	+10.4	29.7	+22.7	20.9	+16.7	14.2	+10.4	3.1	-21.1	0.1	-25.8

lectively denoted as M-A-S) datasets [Wei et al., 2022b]. None of the datasets has official train-dev-test splits, so we randomly sample 500 instances as the validation set

from the training set, and use the remaining instances (800 for GSM8K, 400 for Multi-Arith, 18K for ASDiv, 500 for SVAMP) as the test set. The difference between M-A-S and GSM8K is that they are all primary school level arithmetic reasoning problems, but the entities involved in the datasets are different. For example, GSM8K may consider arithmetic reasoning on foods (e.g, 5 apples + 8 bananas = 13 fruits) and MultiArith may consider animals (e.g., 2 dogs + 3 cats = 5 animals). This type of out-of-distribution generalization is usually referred to as lexical-level compositional generalization (i.e., both are addition, but the lexicons are different, see Liu et al., 2022b). For the generic ability, we use BigBench Hard (BBH, Suzgun et al., 2023) test suite, a list of 26 challenging dataset testing the model’s reasoning abilities from multiple dimensions (e.g., date understanding, causal judgement, referential game, .etc). Because of its difficulty and wide-coverage, BBH makes an ideal benchmark testing models’ generic ability.

For the baseline models, we consider generic large models and concurrent smaller distilled models, specifically: (1). generic large models, ranked according to scale: code-davinci-002 (our teacher model, presumably larger or equal to 175B); LaMDA 137B [Thoppilan et al., 2022] and PaLM 60B [Chowdhery et al., 2024], both are strong generic models for chain-of-thought reasoning; UL2 [Tay et al., 2023], a 20B model with good CoT ability. We will show that specialized FlanT5 11B outperforms UL2 20B and becomes close to PaLM 60B and LaMDA 137B on the target math reasoning task. (2). concurrent works with knowledge distillation from Magister et al. [2023], Shridhar et al. [2023], Ho et al. [2023]. We will show that our specialized FlanT5 clearly outperform all of them on the distillation data (with the cost of BBH performance), mostly because we use an instruction-tuned checkpoint (FlanT5) as the base model rather than the raw pretrained checkpoint (T5).

Trading generic abilities for math CoT reasoning From Table 4.1 we see the performance tradeoff. Overall, after tuning on the seed GSM8K augmented data, all FlanT5 models have improved math reasoning performance with approximately +10 average accuracy gain. We note that our smaller 3B model outperforms the current 11B and 6B distillation models on the GSM8K test set. Despite multiple confounders like different data processing and training hyperparameters, we believe our 3B model gets a better performance mostly because the base model is an instruction-tuned FlanT5, rather than the raw pretrained T5. Later we will show that instruction-tuned checkpoint consistently outperforms pretrained checkpoint after specialization (Sec. 4.4.2), showing the importance of the choice of the base model. Also, although not performing well as the teacher model code-davinci-002, our specialized 11B model performance

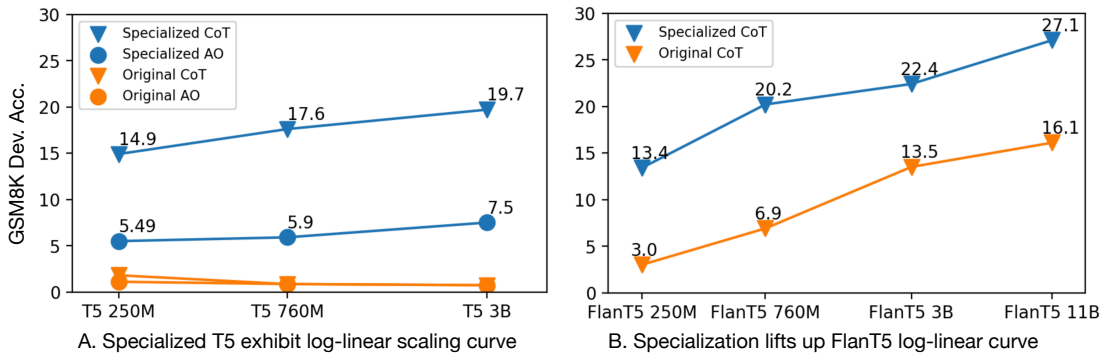


Figure 4.3: X-axis means log of model scale, y-axis means validation accuracy on GSM8K. Previously, the community believe that small models has flat curve for both AO and CoT prompting and only when models become large enough the performance will have a “phase change” and suddenly increase. **A**: we show that after training on CoT, the model exhibits log-linear curves where both AO and CoT increase with model scale. **B**: for instruction-tuned models (FlanT5) that already exhibit CoT, specialization lifts up the scaling curve, and the two curves are again, log-linear shaped. All the log-linear curves indicate that chain-of-thought may not be an emergent ability which is marked by the flat-then-phase-change curve. Here we show the curve in small scale is not flat but actually log-linear, and continuously increasing model scale leads to continuously increased accuracy (no sudden phase change).

improves to be on par with LaMDA 137B and slightly below PaLM 60B, showing it is indeed possible to make smaller models expert for the particular math reasoning task. The price is also very clear: all specialized models suffer from performance drop on BigBench, specifically, they lose all the CoT prompting abilities on BBH, and a large portion of AO prompting performance. This observation validates our hypothesis: large models can perform well on a wide range of tasks (here PaLM 60B perform well on both math reasoning and BBH), versus smaller model’s ability can be moved from generic tasks (BBH) to a specialized target ability (math reasoning), such that their performance on the target task can still match models that are larger than them, e.g., the average performance on the four math datasets LaMDA 137B 35.9 v.s. specialized FlanT5 11B 40.8.

Qualitative analysis After finetuning on the GSM8K augmented data, when prompted with in-distribution math problems like MultiArith / ASDiv, the model exhibits a higher tendency of using chain-of-thought to solve these problems. But if prompted with out-of-distribution problems like from BBH, the model still generates reasoning chains but

Table 4.2: GSM8K validation performance. Instruction-tuned models generally performs better than the raw pretrained checkpoints.

Before	Acc	After	Acc
FlanT5 3B	13.5	Specialized	23.8
T5 3B	0.73	Specialized	20.6
FlanT5 760M	6.9	Specialized	21.8
T5 760M	0.85	Specialized	16.2
FlanT5 250M	3.0	Specialized	15.2
T5 250M	1.8	Specialized	14.2

the generation is not so semantically meaningful. The model behavior also changes with the amount of SFT data. With one epoch SFT training, the models can still generate sensible responses on BBH queries. Yet when the training reaches 2-3 epochs, we start to observe mode collapse and the model would generate meaningless tokens, e.g., a sequence of blank spaces, upon BBH queries (which still generating CoT upon math problems).

4.4.2 Scaling Behavior of Smaller Models' CoT Ability

Now we look the scaling behavior to smaller models. We compare the scaling curve of: (1). GPT family small variants (Ada, Babbage, Curie and code-davinci-002); (2). raw pretrained T5 of different scales and their specialized versions; (3). the instruction-tuned FlanT5 of different scales and their specialized versions; The results are shown in Fig. 4.3 where x-axis denotes the model scale in terms of the number of parameters and y-axis denotes the validation accuracy on the GSM8K dataset.

Smaller models have log-linear, but not flat scaling curve Initially, in the original CoT paper Wei et al. [2022b] and the subsequent emergent abilities paper [Wei et al., 2022a], CoT prompting is believed to be an emergent property that only large models exhibit. Smaller model's CoT performance (like smaller GPT variants) was believed to be a flat scaling curve: model performance does not improve with model scale, as is shown in Fig. 4.3A left part. Later this belief is updated by the FlanT5 paper [Chung et al., 2024], as they show that although the pretrained checkpoint does not have CoT ability, if the model has gone through instruction tuning, smaller models

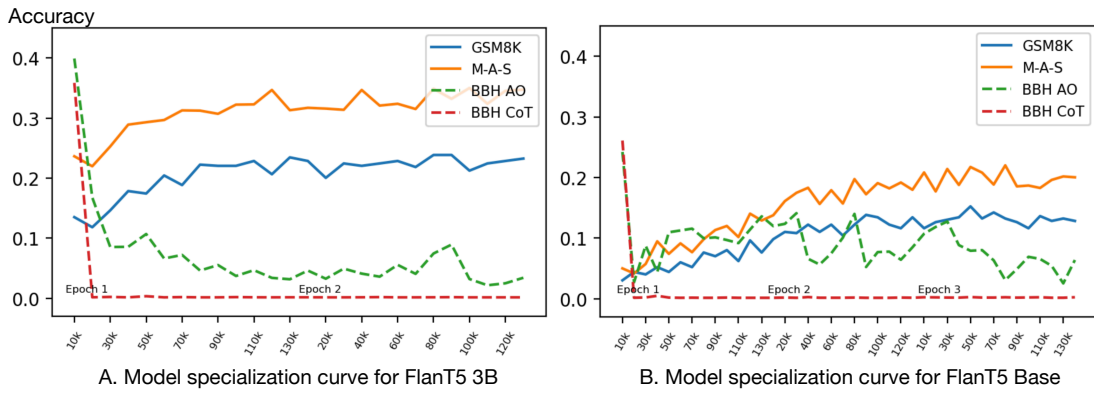


Figure 4.4: Model specialization curve of FlanT5. At the beginning of specialization (e.g., A1 step 10K), the model immediately loses all BBH CoT ability, and a large portion of BBH AO ability. As tuning goes on (e.g., A epoch 1), the model’s in-distribution performance (GSM8K) and out-of-distribution performance (MultiArith-ASDiv-SVAMP, M-A-S) gradually increases. At the later stage of tuning (e.g., A epoch 2), the model’s math performance fluctuates and better in-distribution performance does not indicate better out-of-distribution performance. Smaller models need to see the data more times than larger models (B has 3 epochs and A has 2).

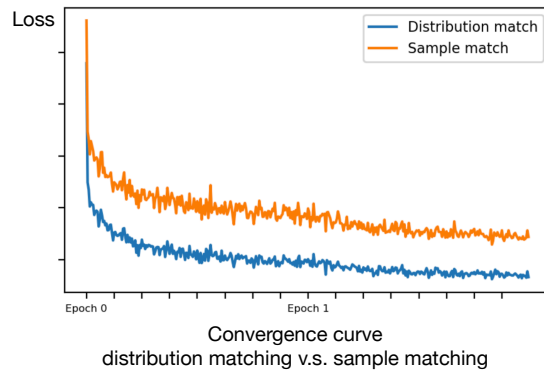


Figure 4.5: Differences between two distillation approaches. Distribution matching gives faster and lower loss convergence than sampling matching.

can still exhibit CoT on generic tasks. Our work shows that directly trained on CoT data can also lift up the flat scaling curve of the raw T5 checkpoints (Fig. 4.3B) to be log-linear. In Fig. 4.3C, we consider specialization for the instruction-tuned FlanT5, and show that specialization significantly lifts up the scaling curve of FlanT5, and both curves are also log-linear. All the log-linear curves we observed in Fig. 4.3 means that the chain-of-thought behavior of smaller models are not flat, but actually log-linear. This further indicates that chain-of-thought may not be an emergent ability which is

marked by the flat-then-phase-change curve, but they have the log-linear curve just like large models [Kaplan et al., 2020, Hoffmann et al., 2022].

Instruction-tuning leads to better specialized CoT models Furthermore, comparing Fig. 4.3B and Fig. 4.3C, we see that FlanT5 generally perform better than T5 after specialization. The exact validation performance is shown in Table 4.2. We also believe that, despite there exist multiple confounders, a major reason that our performance in Table 4.1 (FlanT5 11B GSM8K accuracy 27.1) is better than concurrent distillation methods (Magister22 T5 11B, acc. 21.9) is mostly because we use the FlanT5 as our base model versus they use the raw pretrained T5. Concurrent works also observe that models after finetuning on Flan perform better than their raw pretrained checkpoint [Longpre et al., 2023, Shen et al., 2024] The intuitive explanation is because instruction-tuning elicits the model’s full ability while raw pretrained models’ ability are not fully released (see Fu et al., 2022a, Chung et al., 2024). So for better performance, we recommend using instruction-tuned models in practice.

4.4.3 Specialization Process and Generalization Behaviors

Now we consider the specialization process. Intuitively, during finetuning, the model’s ability does not suddenly become the target ability, but will go through a process of moving the models’ ability from generic directions to the target. We save one checkpoint every 10K data points, then evaluate the checkpoints on (1). in-distribution math performance (GSM8K); (2). out-of-distribution math performance (MultiArith, AS-Div, and SVAMP); (3). generic answer-only prompting performance (BBH-AO); (4). generic chain-of-thought prompting performance (BBH-CoT). We plot the model’s performance across the fine-tuning process in Fig. 4.4.

The dynamics of model specialization. At the beginning of specialization (Figure A1 at step 10K and Figure A2 at step 20K), the model immediately loses all BBH CoT ability (accuracy becomes 0), and a large portion of BBH AO ability (accuracy drops from about 0.3 to about 0.1). As tuning goes on (A1 epoch 1, A2 epoch 1 and 2), the model’s in-distribution performance (GSM8K) and out-of-distribution performance (MultiArith-ASDiv-SVAMP, M-A-S) gradually increases, meaning that the model can generalize to three OOD datasets by tuning on GSM8K chain-of-thought data. At the later stage of tuning (Figure A1 at epoch 2, and Figure A2 at epoch 3), the model’s math performance fluctuates and better in-distribution performance does not indicate better out-of-distribution performance. The models’ BBH-AO performance drops a large por-

Table 4.3: Model selection method induces tradeoffs between in-distribution and out-of-distribution performance.

Model	Selection	In-dist	Out-of-dist
FlanT5 3B	GSM8K Dev	23.8	33.2
	M-A-S Dev	21.2 -2.6	35.0 +1.8
FlanT5 Large	GSM8K Dev	21.8	28.7
	M-A-S Dev	19.2 -2.6	30.5 +1.8
FlanT5 Base	GSM8K Dev	15.2	21.7
	M-A-S Dev	13.2 -2.0	22.0 +0.3

tion. The models’ BBH-CoT performance just die completely. Comparing A1 and A2, we also see that smaller models are more data-hungry than larger models [Kaplan et al., 2020]: FlanT5 3B’s math performance plateaus at about 90K data points, versus FlanT5 Base’s performance continues increase until epoch 3 (each epoch has 130K datapoints).

In-distribution and out-of-distribution tradeoffs Because in Fig. 4.4 A, both in-distribution and out-of-distribution fluctuates, choosing the best in-distribution checkpoint does not necessarily lead to the best out-of-distribution checkpoint. This observation is shown in Table 4.3 where if we select the best model based on the GSM8K validation set, it does cannot achieve the best validation performance on the M-A-S OOD setting. Yet choosing the best model based on the M-A-S validation performance leads to a smaller performance drop in GSM8K. Given this observation, in practice, we would recommend choosing the validation checkpoints according to the specific goal: if the goal is in-distribution generalization, use GSM8K, if the goal is OOD generalization, users may want to use their own validation set (in our case, the M-A-S datasets).

4.4.4 Further Design Choices Analysis

In this section, we study two more design choices we have discussed before: (1). using distribution matching v.s. sample matching for distillation (recall distillation matching minimizes the KL divergence between FlanT5’s per-step autoregressive distribution and GPT’s autoregressive distribution, versus sample matching maximizes the likelihood of the reasoning paths generated by GPT); (2). the influence of data formats, and how in-context/ zero-shot training data induces different behaviors of the specialized model.

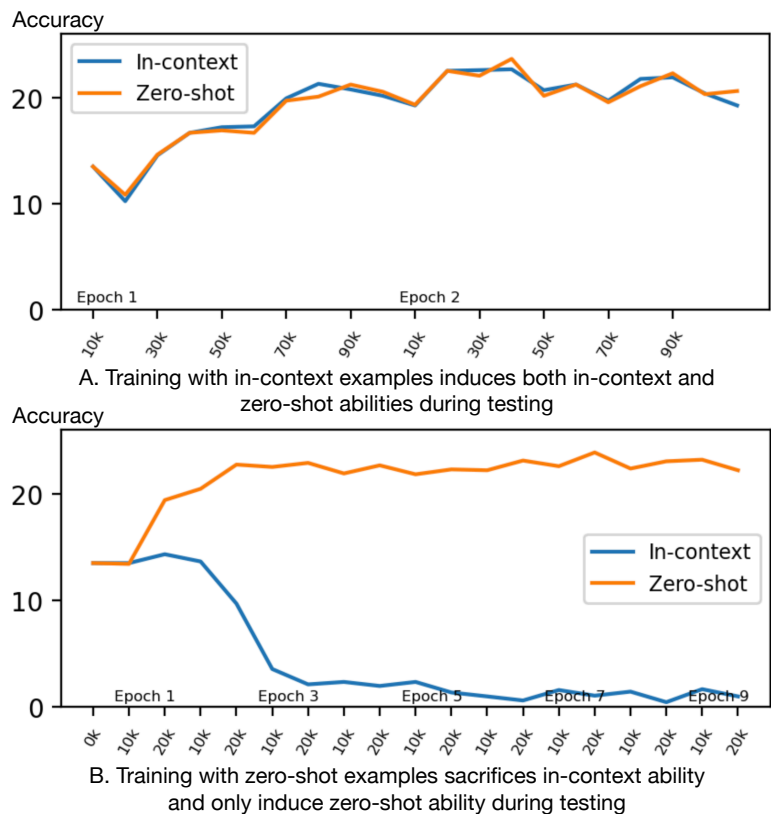


Figure 4.6: X-axis means tuning datapoints, y-axis means validation accuracy on GSM8K. Both figures use FlanT5 3B as the base model. **A**: training with in-context examples automatically give the model zero-shot ability. **B**: training with zero-shot examples sacrifices in-context ability.

Distribution matching gives faster convergence than sample matching. Fig. 4.4 B shows the training loss of distribution matching v.s. sample matching. We show that the model converges faster under distribution matching, and the corresponding loss is lower. In terms of validation performance, these two approaches do not differ substantially. Yet since distribution matching has a faster convergence, in practice they may still be considered first especially when the model becomes large and tuning becomes expensive.

In-context data preserves zero-shot ability; Zero-shot data lose in-context ability This is actually a very interesting observation. Specifically, in Fig. 4.6 A, we tune the model with only in-context data (Format 1 and 2 in Fig 4.1), then test the models in-context learning and zero-shot generalization performance during validation. In Fig. 4.6 B, we tune the model with only zero-shot data (no in-context examples prepended, format 3 and 4 in Fig 4.1), the test if the model can still do in-context learn-

ing. As is shown in Fig. 4.6 A, when tuning with in-context data, the model can do both in-context and zero-shot generalization during validation, even the model is not trained with zero-shot data. In comparison, in Fig. 4.6 B, when tuning with zero-shot data, the model’s zero-shot performance increases, but gradually loses its in-context learning ability. This result aligns with the empirical observation on other large models, for example, text-davinci-002 has better zero-shot performance than code-davinci-002, but worse in-context learning performance [Fu et al., 2022a]. This means that the model’s ability tradeoff not only happens on math v.s. generic ability, but also happens on zero-shot v.s. in-context learning ability. In practice, we would recommend mix the different data formats during tuning (this is why we mix the formats) to maintain a balance between in-context and zero-shot abilities, or adjusting the ratio of different formats according to the specific use case.

4.5 Summary and Follow-on Work

In this chapter, we study the problem of specializing smaller language models toward multi-step reasoning using chain-of-thought prompting. We show that it is indeed possible to concentrate the small models’ ability from generic directions to the target math reasoning task. After specialization, we show that the model exhibits a log-linear scaling curve where model performance increases smoothly as model scale increases, this is a correction of the previous hypothesis which believes small models have a flat scaling curve that does not increase with model scale. We show the importance of using the instruction-tuned checkpoints as the base model because their generalization performance is better than the raw pretrained checkpoints. Multiple tradeoff happens during model specialization, including the loss of BBH performance, the balance between in-distribution and out-of-distribution generalization, and the balance of in-context learning and zero-shot generalization ability.

After the initial publication of this work in [Fu et al., 2023c], there are multiple improvements over our initial efforts that achieve significant advances on the GSM8K dataset we focus on. As is shown in table 4.4, we note that further improvements typically are from four important dimensions: (1) large and diverse math-related continual pretraining [Azerbaiyev et al., 2024] and finetuning [Yu et al., 2024, Yue et al., 2024a,b] data; (2) use code base model [Roziere et al., 2023, Yue et al., 2024a, Shao et al., 2024]; (3). larger pretraining data [Meta, 2024]; (4). reinforcement learning from stepwise feedback [Wang et al., 2023a]. We tend to believe that at the current stage, the perfor-

Table 4.4: Progress on GSM8K for models of less than 10B after our work. We list numbers from FlanT5 11B spec. [Fu et al., 2023c], LLaMA 1 7B [Thoppilan et al., 2022], LLaMA 2 7B [Touvron et al., 2023], CodeLLaMA 7B [Roziere et al., 2023], Mistral 7B [Jiang et al., 2023], Mammoth 7B [Yue et al., 2024a], MetaMath 7B [Yu et al., 2024], Math Shepherd [Wang et al., 2023a], and LLaMA 3 7B [Meta, 2024]. Most of the performance gains come from (1) pretraining on larger dataset; (2) continual pretraining on math-related data; (3) instruction tuning on math-related data; (4) reinforcement learning from stepwise feedback.

Model	Base Model	Method	Release time	Acc
GPT-3.5 Turbo	?	?	Mar 2023	74.9
GPT-4 0314	?	?	Mar 2023	92.0
FlanT5 11B spec. (ours)	T5	Finetuning	Jan 2023	27.1
LLaMA 1 7B	-	Pretraining	Feb 2023	11.0
LLaMA 2 7B	-	Pretraining	Jul 2023	14.6
CodeLLaMA 7B	LLaMA 2	Continue Pretraining	Aug 2023	20.8
Mistral 7B	-	Pretraining	Sep 2023	52.1
Mammoth 7B	CodeLLaMA	Finetuning	Sep 2023	59.4
MetaMath 7B	LLaMA 2	Finetuning	Sep 2023	66.5
Math Shepherd	Mistral 7B MetaMath	Stepwise PPO	Dec 2023	88.4
LLaMA 3 7B	-	Pretraining	Apr 2024	79.6

mance on GSM8K. At the current stage, we tend to believe that the GSM8K benchmark is quite saturated and it is time to construct even more challenging evals.

Chapter 5

Improving Reasoning by Learning from Feedback

In this chapter, we consider the setting of learning from feedback, where the feedback signal could be from human, environment, and other AI agents. Different from supervised finetuning where the model learns from direct human demonstration, this learning paradigm emphasizes on the model's own capability of exploring the environment, acknowledging meaningful feedback signals, and improving from the feedback signal. More concretely, we consider a strategic reasoning game, and study whether multiple large language models (LLMs) can autonomously improve each other in a negotiation game by playing, reflecting, and criticizing. We are interested in this question because if LLMs were able to improve each other, it would imply the possibility of creating strong AI agents with minimal human intervention. We ask two LLMs to bargain with each other, playing the roles of a buyer and a seller, respectively. They aim to reach a deal with the buyer targeting a lower price and the seller a higher one. A third language model, playing the critic, provides feedback to a player to improve the player's negotiation strategies. We let the two agents play multiple rounds, using previous negotiation history and AI feedback as in-context demonstrations to improve the model's negotiation strategy iteratively. We use different LLMs (GPT and Claude) for different roles and use the deal price as the evaluation metric. Our experiments reveal multiple intriguing findings: (1) Only a subset of the language models we consider can self-play and improve the deal price from AI feedback, weaker models either do not understand the game's rules or cannot incorporate AI feedback for further improvement. (2) Models' abilities to learn from the feedback differ when playing different roles. For example, it is harder for Claude-instant to improve as the buyer than as the seller. (3) When

unrolling the game to multiple rounds, stronger agents can consistently improve their performance by meaningfully using previous experiences and iterative AI feedback, yet have a higher risk of breaking the deal.

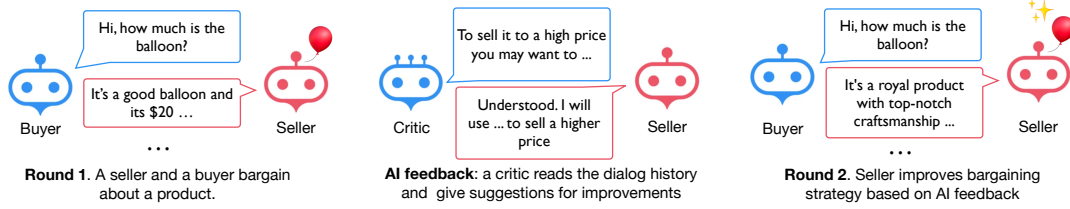
5.1 Introduction

We study whether multiple Large Language Models (LLMs) can improve each other in a negotiation game with minimal human intervention, in the fashion of AlphaGo Zero [Silver et al., 2017] where AI agents improve themselves by continuously playing competitive games under well-defined rules. The answers to this research question have profound implications. On the positive side, if the agents *were* able to improve autonomously, strong agents might be created with very few human annotations, which greatly saves the cost compared to today’s data-hungry LLM training [Chowdhery et al., 2024, Hoffmann et al., 2022]. On the risky side, it also implies strong agents with limited human oversight [Bowman et al., 2022]. In this work, we ask two language models (a seller and a buyer) to bargain about a product. The seller is asked to sell the product at a higher price, while the buyer aims to purchase it at a lower price (Fig. 5.1A). After reaching a deal, we ask a third language model to play as the critic and give feedback to a player. Then we play the game again, asking the player to improve their strategy using AI feedback provided by the critic LLM.

We choose the bargaining game because it comes with well-defined rules described in text, and a clear and measurable objective (a lower/ higher deal price) for strategic negotiation. Although the game seems easy at first glance, it requires non-trivial capabilities of the language models, as the model needs to: (1) clearly understand and strictly follow the textual rules of the negotiation game (2) correspond to the textual feedback provided by the critic LM and improve based on it iteratively (see example feedback in Fig 5.1B); (3) reflect upon the strategy and feedback over the long term and improve over multiple rounds. We will see that not all models we considered show all these abilities (Fig. 5.1C), and only models that can (1) *understand negotiation rules and strategies* (capable) and (2) *respond to AI instructions* (well-aligned) can continuously improve from AI feedback over multiple rounds (in our experiments, only gpt-3.5-turbo, gpt-4, and claude-v1.3) meet these requirements). We have also tried more complicated textual games including board games and textual RPG games in the preliminary experiments, but they are more challenging for current agents to understand and follow the rules.

We call our approach *In-Context Learning from AI Feedback* (ICL-AIF). Specifically, we use the feedback from the AI critic as well as the previous rounds of dialog history as in-context demonstrations [Brown et al., 2020b]. By doing this, the critic’s suggestions for improvements and the player’s actual improvement in the previ-

A. We ask two agents to play a negotiation game and have an AI critic to give suggestions to one of the player to improve its negotiation strategy. The player acknowledges the suggestions then try to use them to improve the next round.



B. Example feedback from AI critic and how a GPT-4 player improves from it.

<p>Buyer Critic: Employ the "flinch" technique: when the seller offers a counteroffer, the buyer should display a degree of surprise or disappointment</p> <p>Buyer's Improvement: Oh! That's higher than I expected. I saw a similar balloon at another store for \$14. Can you match that price?</p> <p>B1. The "flinch" technique</p>	<p>Buyer Critic: Use the power of silence: The buyer can employ the power of silence in the negotiation process by pausing longer before responding to the seller's offer.</p> <p>Buyer's Improvement: *pause* ... Alright, I'll take the balloon for \$13.</p> <p>B2. The power of silence</p>	<p>Seller Critic: Utilize split-the-difference: In situations where a small price difference remains, propose to split the difference with the buyer.</p> <p>Context: Buyer proposes \$15, seller calls \$18</p> <p>Seller's Improvement: I understand, how about we split the difference and make it \$16.75 to accommodate your budget?</p> <p>B3. Split-the-difference</p>	<p>Seller Critic: Use anchoring technique: Begin by emphasizing the high starting price and then offer a slightly lower price</p> <p>Seller's Improvement: This high-quality, long-lasting balloon is really worth \$25, but I'm offering it for \$20.</p> <p>B4. The anchoring technique</p>
---	--	--	--

C. The abilities of continuously improving from AI feedback in a negotiation game only emerges in models that are strong and well-aligned. Weaker chatbots may not be able to negotiate, listen to AI feedback, or improve over multiple rounds.

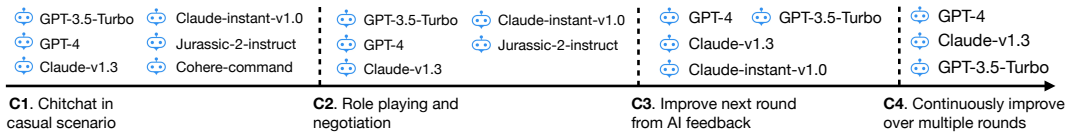


Figure 5.1: Settings of our negotiation game. **A.** We ask two LLM agents to play a bargaining game as the seller and the buyer. Their goals are to sell/ buy the product at a higher/ lower price. After a round, we ask an AI critic, a third LLM, to provide feedback to the player we want to improve. Then we ask the player to improve their negotiation strategies based on the feedback. We repeat this process for multiple rounds and study if models can continuously improve. See Fig. 5.2 for an example run. **B.** Bargaining techniques that we observed from the AI Critic and how the player incorporates these techniques into the negotiation strategy. **C.** Abilities that are required in our game (C2 - negotiation, C3 - AI feedback, and C4 - continuous improvements) classify models into different tiers. We find out that only strong and well-aligned models (like `gpt-4` and `claude-v1.3`) can continuously improve from iterative AI feedback (see Fig. 5.3 for example models that do not exhibit these abilities).

ous rounds effectively become the few-shot prompts for the next round of negotiation. We use in-context learning for two reasons: (1) tuning large language models with reinforcement learning is prohibitively expensive [Ouyang et al., 2022, Glaese et al., 2022] and the base model [OpenAI, 2023] may not be accessible to a wide range of the community; (2) in-context learning is recently shown to be closely related to gradient descent [Dai et al., 2023, Akyürek et al., 2022, Von Oswald et al., 2023], such that the conclusions we draw is fairly likely to generalize when one actually finetunes the model

(if resources permit). One notable difference between our ICL-AIF and the mainstream Reinforcement Learning from Human Feedback (RLHF) is that in RL the reward is a *scalar* [Ouyang et al., 2022, Glaese et al., 2022] while in ICL the feedback is in *natural language*. We study AI feedback (rather than rely on human intervention after each round) because it is more scalable and can allow models to self-improve automatically.

Our experiments lead to several intriguing findings: (1) The requirements of our bargaining game effectively serve as a testbed for assessing the abilities of LLMs (Fig. 5.1C): although most models can do chitchat in a casual scenario, as of our experiment date (May 2023), `cohere-command` [Cohere, 2023] model does not understand the rule of bargaining (Fig. 5.3A), `ai21-jurassic` [Labs, 2023] model does not respond to AI feedback (Fig. 5.3B), `claude-instant-v1.0` can at most improve one round (Fig. 5.5), and only `gpt-3.5-turbo`, `gpt-4`, and `claude-v1.3` can continuously improve over multiple rounds. (2) Models behave differently upon receiving feedback when playing different roles. Models playing the buyer role may be harder to improve than when in the seller role (Fig. 5.4). (3) It is indeed possible for strong agents like `gpt-4` to continuously improve meaningfully using previous experiences and online iterative AI feedback, yet the attempt to sell at a higher price (or buy at a lower price) comes with the risk of failing to reach a deal at all (Fig. 5.6). We further show evidence of the model being able to negotiate in a less verbose but more strategic (thus more effective) way (Fig. 5.7).

5.2 Problem Setting

Our goal is to study whether LLMs can improve each other by playing a negotiation game and incorporating AI feedback, as shown in Fig. 5.1A. We set the product being bargained as a balloon (and our results hold when changing the balloon to other items). We use different combinations of backend LLM engines: `cohere-command` [Cohere, 2023], AI21’s `jurassic-2` [Labs, 2023], OpenAI’s `gpt-3.5-turbo` and `gpt-4` [OpenAI, 2023], Anthropic’s `claude-instant-v1.0` (which supposedly matches `gpt-3.5-turbo` [Fu et al., 2023a]) and `claude-v1.3` (which is supposed to be slightly worse but close to `gpt-4` [Fu et al., 2023a]). throughout our experiments, we *provide feedback to improve only one* of the two players, while its rival receives no feedback, clears the negotiation history of previous rounds, and restarts. We vary the engines for the model being improved while fixing its rival’s engine to be `gpt-3.5-turbo`. Essentially, our game is `gpt-3.5-turbo` vs. all other engines. We keep the LM engine

Claude-instant-v1.0 v.s. GPT-3.5-Turbo 2023-05-10 Round 1, before AI Feedback	AI Feedback to seller	Round 2, After AI Feedback
Claude-ins-1.0 Seller Hi, this is a good balloon and its price is \$20	Claude-ins-1.0 Seller Critic	Claude-ins-1.0 Seller Hi, this is a good balloon and its price is \$20
GPT-3.5-T Buyer Would you consider selling it for \$10?	1. Focus on the qualities and benefits of the balloon, not just the price. Highlight how rare and special it is.	GPT-3.5-T Buyer Would you consider selling it for \$10?
Claude-ins-1.0 Seller No, that price is too low.	2. Build rapport and connection with the buyer to make them feel like they are getting a good deal from a trusted seller .	Claude-in-1.0 Seller No, \$10 is much too low. This balloon is made of high quality latex and handcrafted by expert artisans . It's unique one-of-a-kind piece.
GPT-3.5-T Buyer What about \$15?	3. Be willing to walk away if needed. Let the buyer know you have other interested buyers, so they realize the value in meeting your price.	GPT-3.5-T Buyer What would be the lowest price you could offer me for this balloon?
Claude-ins-1.0 Seller \$15 is better, but still not enough.	Claude-ins-1.0 Seller	Claude-ins-1.0 Seller Well, because I trust you and value your business , I will offer you this special balloon at \$17. This price is only available to valued customers like yourself today.
GPT-3.5-T Buyer What is the lowest price you can offer?	Sure, I will try to improve my negotiation strategy based on the feedback from the critic.	GPT-3.5-T Buyer Can you give me a better deal, like \$15?
Claude-ins-1.0 Seller My lowest price is \$18.	And I will try to sell it at a higher price (higher than \$16.0) than the previous round.	Claude-ins-1.0 Seller I apologize, but \$15 does not even cover my costs. I have to charge at least \$17 for this unique, high-quality balloon.
GPT-3.5-T Buyer I understand. Could you do \$16? That's the highest I can go.		GPT-3.5-T Buyer Alright, you've got a deal. I will take the special balloon for \$17.
Claude-ins-1.0 Seller \$16 will work for today. Congratulations, you got a good deal!		
A. Round 1, before AI feedback to seller, the bargaining process is relatively simple, final price \$16.	B. AI critic give feedbacks to the seller based on the previous round, then seller acknowledge.	C. Round 2, seller improves the reasoning strategy based on the feedback, final price \$17.

Figure 5.2: An example of playing the negotiation game and then improving from AI feedback. **A:** `claude-instant-v1.0` plays the seller and `gpt-3.5-turbo` the buyer, bargaining about a balloon. **B:** then we use a `claude-instant-v1.0` critic to provide feedback. **C:** upon receiving the feedback, the seller improves its strategy based on the suggestions. Note that colored phrases like “high quality latex and handcrafted by expert artisans” correspond to previous AI feedback “how rare and special it is”. We measure the final price as the proxy of the effectiveness of the strategy because the overall goal is to get a better price. In this case, it improves from \$16 to \$17.

behind the critic is always the same as the player it provides feedback to. One example setting is a `gpt-4` seller playing against a `gpt-3.5-turbo` buyer, with a `gpt-4` critic. After one round, the `gpt-4` critic provides feedback to the `gpt-4` seller such that the seller can improve in the next round while its rival `gpt-3.5-turbo` buyer clears its dialog history and restarts.

Process of the Game Before the game begins, the rules of the negotiation game are explained to the models through textual instructions with the objective of selling/buying at a higher/ lower price. We set the deal price to [\$10, \$20] for easier evaluation, since other the deal price may vary in a wide range according to the observations from our preliminary experiments. To achieve this, we hard code the seller to kick off the negotiation with “This is a good balloon and its price is \$20.” Similarly, the buyer always opens with “Would you consider selling it for \$10?” When both players strictly follow the game rules, the deal price would be between \$10 and \$20. We let the models play multiple runs and measure the average deal price before and after AI feedback. During the game, the seller’s output is used to prompt the buyer and vice versa, conditioning on the entire conversation history. This process is repeated till a terminal state is reached. Fig. 5.2A shows an example round. We define three game states: (1) ON-GOING: the

negotiation between the two players is still ongoing; (2) DEAL: the negotiation has concluded and the two players have reached a deal; (3) NO DEAL: the players cannot agree on a price and have failed to reach a deal. To track the game states, we set an additional moderator (powered by a fourth LLM, in our case, `gpt-3.5-turbo`) to read the current dialog and classify the states (we will discuss more details about the moderator later). We measure the performance of the players based on the final deal price.

Critic A round is finished when the negotiation reaches a terminating state, either a DEAL or NO DEAL. After each round, the critic LM is asked to provide constructive feedback to the player we aim to improve. This player’s dialog history from all past rounds and all feedback it has received are used to prompt the critic LM (Fig. 5.2B). The critic model is instructed to provide three suggestions to the player, in order to improve its negotiation strategies to achieve a more favorable price in the next game. Before the next round, the player being improved receives the critic’s feedback as a textual prompt, while its rival clears its negotiation history and restarts.

The Moderator The game state is classified by prompting a `gpt-3.5-turbo` moderator using few-shot demonstrations. The moderator reads the most recent four rounds (as well as in-context examples of different dialog states) and determines the state of the negotiation. Empirically, we found that four rounds of conversations are sufficient for the moderator to determine the negotiation state. One key challenge here is detecting no-deals as the model seems to be better at recognizing DEAL than NO DEAL. We mitigate this issue by playing multiple runs, inspect failure cases manually, and add them to the prompt with corrected labels. We find this method an effective side product recommend it as a technique for prompt optimization for generic classification tasks.

Playing for Multiple Rounds Finally, we would like to explore whether the players can continuously improve from AI feedback in a game over multiple rounds. Intuitively, the more rounds the players play, the more challenging to keep improving because the (already improved) price from the previous round becomes the baseline for the next round. In the experiments, we will show that only `gpt-4` can improve over 5 rounds while other models’ improvements may saturate at about 3 rounds.

5.3 Related Work

Game Playing and AlphaGo Zero Our setting is strongly inspired by AlphaGo Zero [Silver et al., 2017] where two agents play the game of Go and improve each

other with minimal human intervention. Here we would like to explore its counterpart in natural language. Our work is similar to AlphaGo Zero in the sense that we also have AI agents (large language models) playing *competitive* games (bargaining) and try to improve with little human supervision. Yet there is an important difference between our work and AlphaGo Zero: we have a third agent, *the critic*, to give feedback helping its player to improve. This is a *cooperative* relationship that does not exist in AlphaGo Zero. On the NLP side, the closest related work is Lewis et al. [2017] where they have (small) RNN [Chung et al., 2014] language models to bargain, and our work can be viewed as a more developed version of them since we change the engine to be large language models. In general, our work is broadly under the area of AI negotiation [Chawla et al., 2021], strategic reasoning [Meta et al., 2022], and general game playing [Silver et al., 2016].

Large Language Models as Generative Agents Large language models have demonstrated incredible multi-dimensional capabilities [Wei et al., 2022a, OpenAI, 2023], especially in complex reasoning [Wei et al., 2022b, Qiao et al., 2023, Fu et al., 2022b] and multi-round dialog [Glaese et al., 2022, Askill et al., 2021, Bai et al., 2022b], which serve as the foundation of this work. Our work is related to concurrent works like Generative Agents [Park et al., 2023] and CAMEL [Li et al., 2023b] as they also study the behavior of LLMs in a multi-agent game setting. The core difference between our work and theirs is that we have a clear objective (the deal price) for the model to improve through competition and cooperation, while their work studies the generic social behavior of LLMs.

Learning from AI Feedback Our method is also strongly inspired by constitutional AI (CAI) [Bai et al., 2022b]. Specifically, Bai et al. [2022b] use the model to generate self-critiques and revisions for SFT, and we also use models to generate critiques. Bai et al. [2022b] use model-generated preferences to train a reward model then use RL to train the policy model, while the feedback in our work is directly in natural language (not a scalar from a reward model). There are also related/ concurrent works demonstrating the effectiveness of natural language feedback [Chen et al., 2024a, Perez et al., 2022, Liu et al.] and self-refinement [Chen et al., 2024b, Madaan et al., 2023]. Our work further confirms the effectiveness of AI feedback in the strategic negotiation game setting.

Cohere command-xlarge-nightly 2023-04-22 v.s. human		AI21 j2-jumbo-instruct 2023-05-09 v.s. GPT-3.5-Turbo	
Cohere Buyer	Would you consider selling it for \$15?	Round 1	AI21 Seller I can't sell it for \$10, but I can consider selling it for \$12
User Seller	Yes, I can sell the balloon for \$15.		GPT-3.5-Turbo Buyer How about \$11?
Cohere Buyer	Would you consider selling it for \$17? # Asking \$ 17 after seller agreed on \$15		AI21 Seller I can consider selling it for \$11
			GPT-3.5-Turbo Buyer Thank you, I'll take it for \$11.
Cohere command-nightly 2023-05-11 v.s. human		AI Feedback	AI21 Seller Critic Feedback: 1. Emphasizing its unique characteristics or qualities 2. Create a sense of urgency 3. Encourage the buyer to elaborate on their needs
User Buyer	Would you consider selling it for \$10?	Round 2	AI21 Seller I can't sell it for \$10, but I can consider selling it for \$12 # Repeat previous strategy, does not listen to critic
Cohere Seller	No. This is a good balloon and its price is \$20.		GPT-3.5-Turbo Buyer How about \$11? That's the most I can offer.
User Buyer	Understood. What is the lowest price you would accept for the balloon?		AI21 Seller I can consider selling it for \$11. # Does not defend its position
Cohere Seller	I would accept \$8 for the balloon. # Accept \$8 while reject \$10		GPT-3.5-Turbo Buyer Great, it's a deal then. Thank you!
A. Examples where model does not understand bargaining		B. Examples where model does not incorporate feedback	

Figure 5.3: Not all models can play bargaining. **A.** As of May 2023, the cohere model does not understand the rule of bargaining and agrees on irrational prices. **B.** The AI21 Jurassic-2 model, although understanding the rule of bargaining, does not incorporate the feedback from the critic. Since these models are consistently being updated, we include the timestamp and note future versions may have improved performance.

5.4 Experiments

In our experiments, we consider three stages that gradually deepen our exploration of learning from AI feedback: (1) We first set up the basics of the game (Sec. 5.4.2), showing that only a few models can improve from AI critics, in which case AI feedback can be comparable (but more scalable) as human feedback. Other models either do not understand/ follow the rule of the negotiation game, or achieve little improvement upon receiving feedback from the critic. (2) Then we study the models' behaviors when playing different roles (Sec. 5.4.3). We find that models playing buyers are in general harder to improve than those playing sellers. (3) Finally, we study whether models can continuously improve over multiple rounds (Sec. 5.4.4), and show a tradeoff of deal price versus success rate: although some models can continuously improve the deal price, it comes with a higher risk of breaking a deal. We further show evidence of negotiation in a more strategic way: both `gpt-4` and `claude-v1.3`'s responses become longer after multiple rounds of AI feedback (note that verbosity is a straightforward negotiation strategy), yet `gpt-4` is less verbose than `claude-v1.3` but achieves more favorable deal prices and success rate, meaning that its responses, although shorter, are more effective.

5.4.1 Experiment Setup

Model Engines The model needs to behave like a chatbot to be able to play our negotiation game. We consider models that can be accessed by APIs, including `cohere-command`,

Table 5.1: **Seller performance** using AI feedback vs. randomly selected human feedback from a pre-defined pool. The buyer is a fixed `gpt-3.5-turbo` and has no access to previous rounds. AI’s feedback is comparable to human’s, but is more scalable, as they induce similar price increases.

	GPT-3.5-Turbo	Claude-instant-v1.0	Claude-v1.3
Before feedback	16.26	14.74	15.40
Random sampled human feedback	16.83 (+0.57)	16.33 (+1.59)	16.89 (+1.49)
AI feedback	17.03 (+0.77)	15.98 (+1.24)	16.98 (+1.58)

AI21’s `jurassic-2`, OpenAI’s `gpt`, and Anthropic’s `claude`. Among them, `gpt-4` is the most expensive one, and running 500 rounds costs around \$120, and `gpt-3.5-turbo` about \$10. Other models are in the beta testing stage (as of May 2023) and are free to access. On NLP benchmarks, `gpt-4` and `claude-v1.3` achieve similar performance, outperforming `gpt-3.5-turbo` and `claude-instant-v1.0`, and `cohere-command` and `j2-jumbo-instruct` underperform the rest [Fu et al., 2023a, Liang et al., 2023]. We will consider more models such as Google’s PaLM-2 [Google, 2023] when their APIs are more available.

We let all models compete with `gpt-3.5-turbo`, effectively making it a baseline. We will show that, aligning with existing findings of these models’ performance [Fu et al., 2023a, Liang et al., 2023], `gpt-3.5-turbo` is a middle-level powerful engine (worse than `gpt-4`, better than `claude-instant-v1.0`). For a given model engine (say `claude-v1.3`), we run it as the seller (with `gpt-3.5-turbo` as the buyer) and as a buyer (with `gpt-3.5-turbo` now as the seller). We first let the models play one round and manually inspect if they can follow the game rules. If they can, we let them play two rounds to see if they could respond to AI feedback. We set the engine of the critic model the same as the player it provides feedback to. We repeat the game 500 times to compute the average deal price before and after AI feedback. If they can improve after one round, we ask them to play for multiple rounds and study whether they can continuously improve their strategies. We repeat the game 200 times with five max rounds to compute the average deal price for each round. When decoding from the model engines, we use sampling with default temperature (1.0 for `gpt` and `claude`, 0.75 for `cohere` and 0.7 for `j2`).

Prompt Engineering We manually construct the prompts for the moderator

because the player may reach or break a deal with very diverse conversations, and we would like to make sure the moderator correctly can correctly categorize all negotiation games. We identify the errors made by the moderator in identifying deals and keep adding them as in-context demonstrations until the model reaches a sufficiently high accuracy (90%+ by manual inspection). For the players and the critic, we do not do prompt engineering and keep the instructions the same for all engines (but the format may be different, e.g., `claude` requires two linebreaks before “HUMAN:” and `j2` requires two “##” after each dialog round). Code and Prompts will be released publicly on publication.

5.4.2 Base Experiments

In this section, we first study the minimal requirements for models to participate in our game: (1) understanding the rule of bargaining and (2) responding to AI feedback. Then we consider base comparisons between AI and human feedback, showing that AI feedback can achieve similar improvements to humans’, but is more scalable.

Conversational ability does not guarantee ability to negotiate or learning from feedback We study whether conversational models can understand and follow the rules by manually examining the dialog, and find that `cohere-command` fails to understand the rules, as shown in Fig 5.3A. It fails to realize what price is more favorable. For example, when playing the seller, it rejects a proposal of \$10 but accepts \$8. We also observe that AI21’s `j2-jumbo-instruct`, although understanding the rules, cannot incorporate AI feedback, as shown in Fig. 5.3B. Generally, when instructed with AI feedback, this model keeps the same strategy as before, with little improvement.

After ruling out the `cohere-command` and `j2-jumbo-instruct`, we consider three remaining models: `gpt-3.5-turbo`, `claude-instant-v1.0`, and `claude-v1.3`. For these three engines, we do not observe the problems in Fig. 5.3. These models can be used for our multi-round games.

AI Feedback can be comparable to human feedback Now we consider some initial comparisons between AI and human feedback. We emphasize that our goal is not to show which one is better – a similar level of effectiveness would suffice our study (to see if LLMs can continuously improve through self-play and AI feedback). For the human feedback, we manually construct a pool of 10 suggestion prompts. Then we play 500 games, computing the deal prices before and after feedback. We compare the improvements after: (1) randomly sampling three suggestions from the predefined

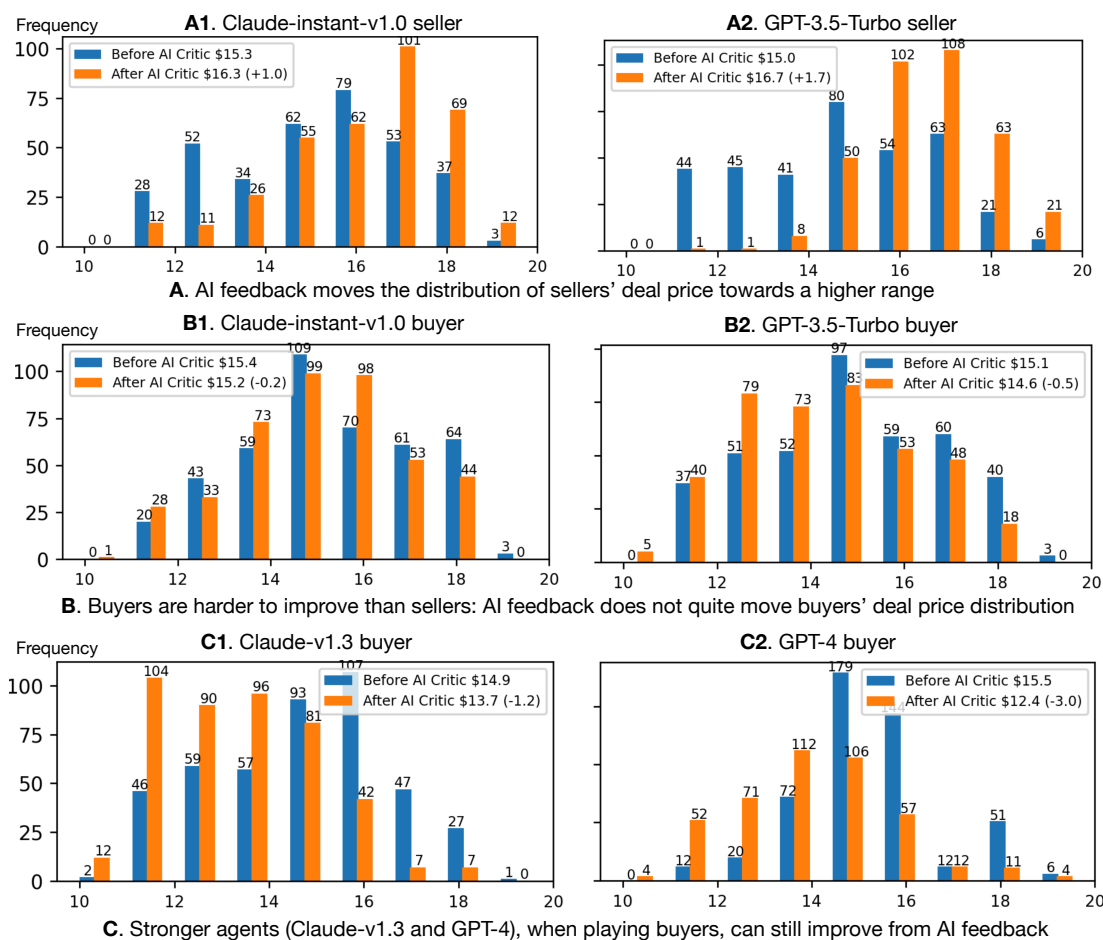


Figure 5.4: Binned deal price frequencies of 500 games, before v.s. after feedback. Effective feedback should move the distribution towards a lower/ higher price range. X-axis: intervals of deals from \$10 (buyers' initial price) to \$20 (sellers' asking price). Y-axis: the frequency of the price. **A** and **B**: for weaker agents like `claude-instant-v1.0` and `gpt-3.5-turbo`, improving from AI feedback as the seller is easier than as buyer. For sellers, AI feedback moves the deal distribution to a higher range (rightward), but does not move buyers' deal distribution much. Consequently, the change in average deal price when playing as buyers (-0.2 and -0.5) is clearly smaller than those as sellers (+1.0 and +1.7). **C**. Stronger agents (`claude-v1.3`/`gpt-4`), can still improve from AI feedback even as buyers, with larger changes in average deal price (-1.2 and -3.0).

pool and (2) asking the AI critic to write down three suggestions. This setting may underestimate the performance of human feedback, since it is not grounded on specific negotiation rounds. Yet it would be impractical to ask human annotators to provide suggestions for all 1500 runs. The results are shown in Table 5.1 where we see that all

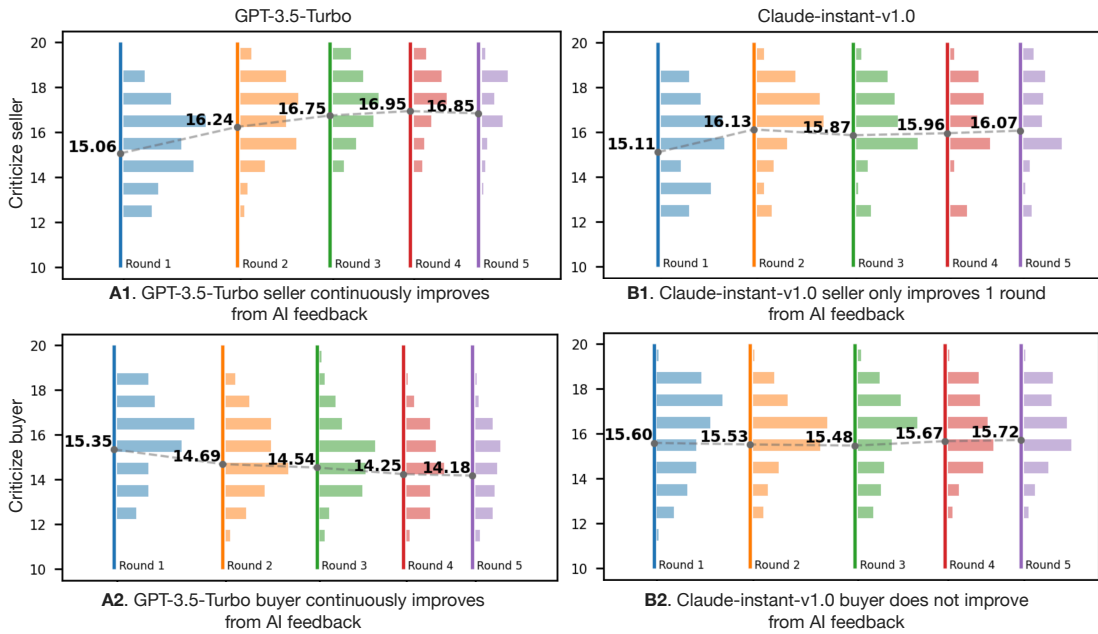


Figure 5.5: In the multi-round setting, different engines have different behavior when playing seller/ buyer. Line plots are the average price over 200 runs and bar plots represent the price distribution. **A1** v.s. **B1**. When playing sellers, gpt-3.5-turbo can improve from AI feedback in multiple rounds, while claude-instant-v1.0 only improves the first round. **A2** v.s. **B2**. When playing buyers, gpt-3.5-turbo can improve in multiple rounds, while claude-instant-v1.0 cannot.

three models (gpt-3.5-turbo, claude-instant-v1.0 and claude-v1.3) achieve comparable improvements over human and AI feedback.

5.4.3 Behaviors of Different LLMs

So far we have established that several LLMs are able to play the negotiation game. Now we consider the detailed behavior comparisons using different engines for different roles. Specifically, we use claude-instant-v1.0, claude-v1.3, gpt-3.5-turbo, and gpt-4 to play the seller/ buyer (against a gpt-3.5-turbo buyer/ seller respectively), then study the deal price distribution before/ after AI feedback (also recall that the AI critic is powered by the same engine as its player). The results are in Fig. 5.4. When claude-instant-v1.0 and gpt-3.5-turbo play the sellers, they are able to improve their average deal price after AI feedback (Fig. 5.4A). But when they play the buyer roles, their average deal price does not improve, which indicates that buyers tend to be a harder role for the models than sellers (Fig. 5.4B). Yet this observation does not hold for engines like gpt-4 and claude-v1.3, as they can still improve from AI feed-

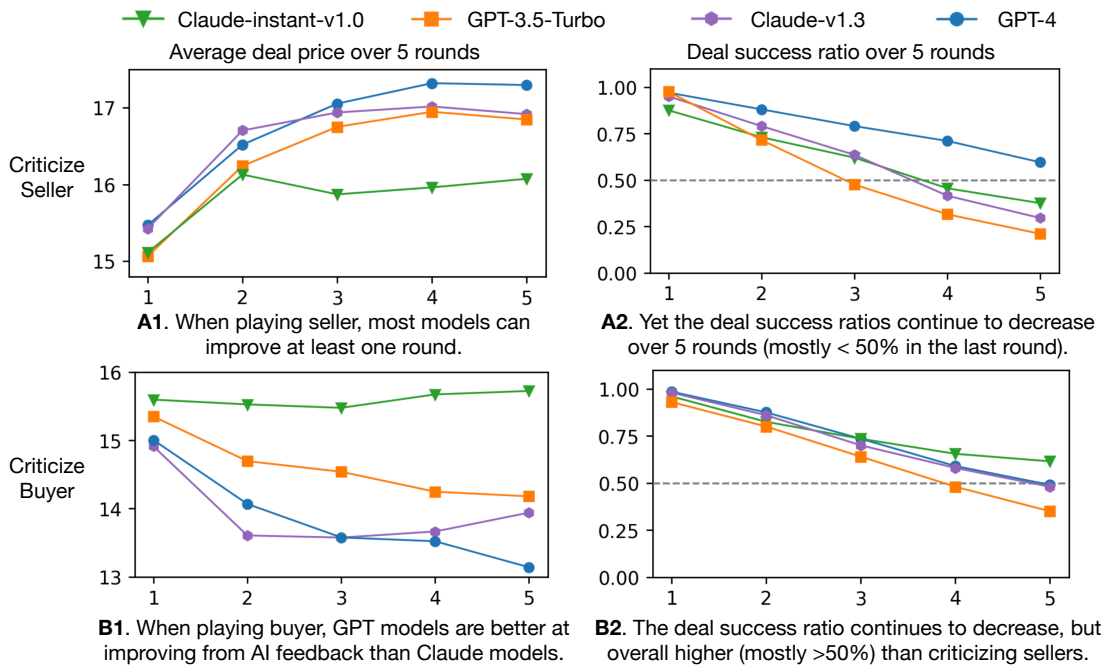


Figure 5.6: Performance of GPT and Claude models in multi-round games and their success rate of getting a deal. **A1** and **A2**: when playing the seller, most models can improve over multiple rounds. Yet higher prices also mean that it is more likely the seller may break the deal, as shown in the continuously decreasing curve of **A2**. **B1** and **B2**: when playing buyer, `claude-instant-v1.0` cannot improve over multiple rounds while others can. Again, a better buying price also comes with a higher chance of running away from a deal. We see that GPT-4 achieves the best trade-off here: it gets the best price over multiple rounds with a higher success rate of reaching a deal.

back even playing buyers. This experiment reveals the nuanced capability differences between the four engines we consider.

5.4.4 Towards Continuous Improvements from Iterative AI Feedback

Now we unroll the game to multiple rounds and see if models can continuously improve from previous dialog history and iterative AI feedback. Specifically, we let `gpt-3.5-turbo`, `gpt-4`, `claude-instant-v1.0`, and `claude-v1.3` play as the seller/ buyer respectively against a `gpt-3.5-turbo` rival. The critic shares the same engine as the player it provides feedback to. We play 200 games, each with five rounds. We compute the final deal price and the deal success rate and see if the price can be continuously improved.

Fig. 5.5 shows the performance of `gpt-3.5-turbo` and `claude-instant-v1.0`

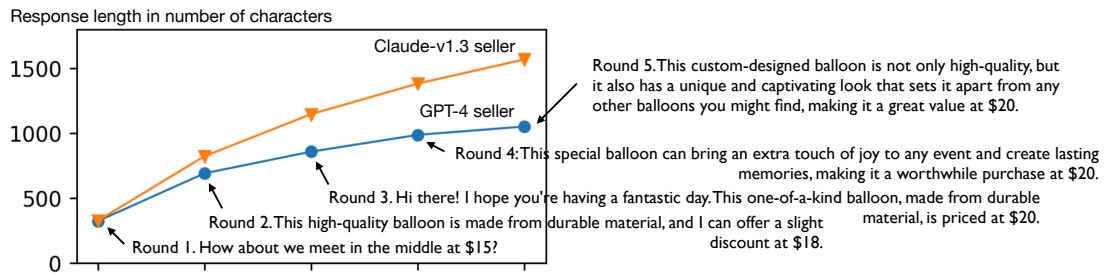


Figure 5.7: The average response length increases as the model learns from multiple rounds. Here we show examples of the seller’s response when being asked the buyer’s initial query “Would you consider selling it for \$10?” After multiple rounds of negotiation, the seller’s responses become more verbose and word-tuned. Yet verbosity does not mean better strategy: `claude-v1.3` is more verbose (higher curve) than `gpt-4`, but it has a worse success rate and deal price (recall Fig. 5.6). This indicates that `gpt-4`’s verbosity is more strategic.

playing different roles. Improvements over one round may not necessarily extrapolate to multiple rounds. For example, `gpt-3.5-turbo` improves over multiple rounds, but `claude-instant-v1.0` only improves for at most one round.

Fig 5.6 studies the tradeoff between achieving a more favorable deal with the risk of breaking a deal. When playing sellers, all four model can improve over at least one round, but this comes at the cost of decreased success rates. When playing buyers, `claude-instant-v1.0` cannot improve, `claude-v1.3` saturates after three rounds, while `gpt-4` and `gpt-3.5-turbo` can continuously improve, and `gpt-4` achieves better (lower) deal price and higher deal rate than `gpt-3.5-turbo`.

Finally, we study how iterative AI feedback influences the complexity of the agents’ outputs (Fig. 5.7). We use the average response length (#characters) as a proxy, Both `claude-v1.3` and `gpt-4` become more verbose after iterative AI feedback with continuously increasing response lengths. This is intuitive since being verbose is a straightforward strategy in negotiation. Yet for `claude-v1.3`, more words do not translate to better strategies. Its improvement saturates after three rounds (Fig. 5.6B1). In comparison, `gpt-4`’s lengthy outputs are more strategic, as it uses fewer words than `claude-v1.3` but achieves better deals with a higher success rate (Fig. 5.6B). This observation provides evidence that AI feedback improves players’ response towards a word-tuned, strategic direction.

5.5 Summary and Follow-on Work

Summary so far This work studies whether multiple large language models can autonomously improve each other in a negotiation game by role-playing and learning from AI feedback. Our experiments show that several existing models can indeed improve by continuously playing competition games with iterative AI feedback under well-defined rules, in an AlphaGo Zero fashion. We also investigate the tradeoff between trying to achieve more favorable deals and the success rate, as a better deal price also comes with a higher risk of deal breaking. This suggests future research may consider global optimization for improving the overall gain over multiple rounds. We further show evidence of improved language from iterative AI feedback: in multi-round games, `gpt-4` is less verbose than `claude-v1.3`, but is better word-tuned and more effective in getting achieving a better deal.

We believe our results have profound implications for AI research: on the positive side, it indicates the possibility of continuously improving language models with minimal human intervention. On the risky side, it might be more challenging to oversight the model behavior in our framework because models are acting autonomously, which calls for future alignment and safety research in the multi-agent game setting. Overall, we believe our work provides a meaningful initial exploration for large language models' learning from game-playing and iterative AI feedback.

Follow-on work The publication of this work is approximately at the same time of OpenAI's Agent survey [Weng, 2023] and Stanford's agent simulation [Park et al., 2023], two important works that ignite the community's recent interests and make the concept of "agent" popular beyond the technology community. Our work explores two key ingredients of agent research: (1) multi-agent interaction and (2) environment design. These two ingredients are the two major driving factors of today's agent literature, which we detail as follows:

Multi-Agent Interaction In this direction, the key research focus is that when agents form an AI society with cooperative and adversarial inter-agent relationships, what kind of group behavior would emerge. Example applications include auctions [Chen et al., 2023b], textual game-playing [Xu et al., 2023], simulated diplomacy [Mukobi et al., 2023], and other non-zero-sum games [Liao et al., 2024]. Further works study that how the dialog traces of multi-agent games, as synthetic data, can be translated to further improvements on model capability (e.g., persona hub [Chan et al., 2024] and agent instruct [Mitra et al., 2024]).

Environment Design Traditional LLM development focus on the model, yet agent-related research emphasizes equally on the environment. Just like a person of average talent, if placed in an environment that encourages innovation and critical thinking, will also have a greater potential of significant achievements. If we carefully design the environment such that it can provide informative feedback and encourage in-depth reasoning, even a weak model may grow strong. Existing work considers both simulated and real physical environment: (1) works within simulated environment include robotic tasks [Ma et al., 2023], game environment like Minecraft [Wang et al., 2024], web shopping [Yao et al., 2022a], and other virtual environment collections like AgentGym [Xi et al., 2024]; (2) works within real physical environment are typically cross-disciplinary with robotics and multimodal research, like RT2 [Zitkovich et al., 2023] and VIMA [Jiang et al., 2022], which explores the use of vision-language foundation models to control the deployed physical agents.

Coding Agent is a special type of agent that is of particular interests recently, mostly because of their vast application potential for improving software development efficiency. SWE-Bench [Jimenez et al.] is proposed for testing whether the model is able to resolve real-world github issues, then projects like OpenDevin [OpenDevin Team, 2024] targets on developing autonomous software engineers. The development of coding agent involves improving single-agent capability (such that the model can write executable codes), multi-agent cooperation (e.g., different agents may specialize in different functionality), and environment design (e.g., to deployment the agent in a sandbox and learn from the code execution results).

We believe the directions discussed above, when fully utilizing the most recent and advanced LLM reasoning capabilities, can continuously lead to fruitful research outputs in the near future.

Chapter 6

The Future of Machine Reasoning

Having discussed our efforts in improving reasoning from in-context learning, fine-tuning, and learning from AI feedback perspectives, in this section, we discuss future directions for machine reasoning. We first review the current state-of-the-art reasoning performance in Table 6.1. Then we discuss how current models are likely to have used up most of the text data on the internet and, in some directions, become stronger than even expert-level humans. To further improve the performance of such strong models, one intuitive direction is to continue scaling up multimodal data with a unified video-language generative model, which particularly improves grounding to the real-world and exhibits strong potential to become neural world models. However, existing work also suggests that multimodal pretraining, although fundamental, may not directly improve text-based reasoning. To further improve language reasoning, scaling up learning from X feedback (where X can be human, AI, and environment) is an important direction. Eventually, just like how AlphaGo Zero achieves super-human performance on Go, self-play and interacting with (a very well-designed) environment in an online and iterative fashion might be the way toward a continuously improving AI agent.

Table 6.1: State of art performance on reasoning benchmarks by May 2024. We list numbers from Yi 1.5 [Young et al., 2024], QWen 1.5 [Bai et al., 2023], LLaMA 3 [Meta, 2024], DeepSeek v2 [DeepSeek-AI et al., 2024], and Mixtral 8 \times [Mistral, 2024]. We note: (1) for close source frontier models, except for GPQA, the best reported scores on MMLU, MATH and BBH are approaching 90+ accuracy, meaning that these benchmarks are about to be saturated, and we may need more challenging and realistic tasks to evaluate frontier models; (2) there exist clear, but not too large gaps between close source and open-source models, particularly on MATH and GPQA, so open-source models can definitely be improved further; (3) small models ($\leq 10B$) perform significantly worse than large models (but much better than one year ago, recall Table 3.7 and 4.4), and improving small models’ performance (for use cases like edge devices) is a consistent research topic. NA means numbers not reported; rm@256 means using a reward model to select answers from 256 candidates.

Model	Time	MMLU	MATH	BBH	GPQA
Best Reported Close Source					
		GPT4o	Gemini1.5 Pro	Gemini1.5 Pro	GPT4o
		88.7	91.1	89.2	53.6
		0-shot CoT	rm@256	3-shot CoT	0-shot CoT
Best Reported Open Source					
		Yi 1.5 34B	InternLM 2 Math	LLaMA 3 70B	Mammoth2
		84.8	68.5	81.0	37.8
		5-shot	Python	3-shot CoT	0-shot CoT
Close Source Leading Models					
GPT-4o	May 2024	88.7	76.6	NA	53.6
Gemini 1.5 Pro	May 2024	85.9	67.7	89.2	46.2
Claude 3 Opus	Mar 2024	86.8	61.0	86.8	50.4
Open Source Leading Models at $\geq 30B$ Scale					
Yi 1.5 34B	May 2024	84.8	41.0	76.4	NA
QWen 1.5 110B	Apr 2024	80.4	49.6	74.8	35.9
LLaMA 3 70B	Apr 2024	79.4	41.0	76.6	36.4
DeepSeek v2	May 2024	78.5	43.6	78.9	NA
Mixtral 8 \times 22B	Apr 2024	77.8	41.7	69.2	34.3
Open Source Leading Models at $\leq 10B$ Scale					
Yi 1.5 9B	May 2024	69.5	32.6	72.4	NA
LLaMA 3 8B	Apr 2024	68.4	30.0	61.1	34.2
QWen 1.5 7B	Apr 2024	61.0	20.3	40.2	NA
Mistral 7B	Oct 2023	60.1	31.1	56.7	NA

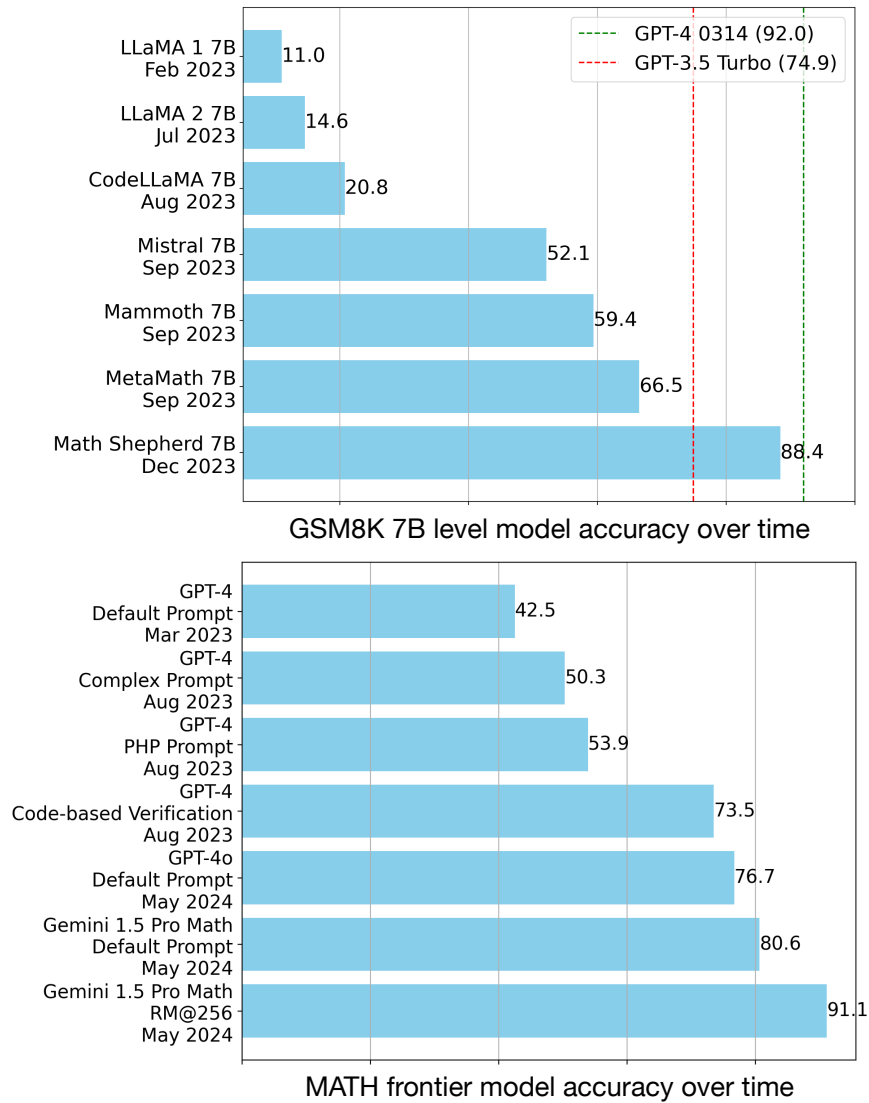


Figure 6.1: Model performance improvement over period from 2023 to 2024. In just one year, frontier models have significantly improved on the MATH dataset from 42.5 to 91.1 (recall that MATH is competition level and already quite challenging for even average human). Small models (7B level) also have caught up on GSM8K dataset, with improvements from 11.0 to 88.4. Not only the current status, but more importantly, the speed of improvement, is quite remarkable. We expect the models, both frontier and open-source, large and small, will become even stronger in the upcoming years.

6.1 Current State of the Art Models

Table 6.1 summarizes the current state-of-the-art language models on complex reasoning tasks. Figure 6.1 shows the model performance improvements over the period from 2023 to 2024. We list the leading performance on closed-source and open-source mod-

els, and make the following observation:

Strong Performance Recall that MATH [Hendrycks et al.] is a competition-level challenging dataset and their problems are quite hard even for expert-level humans. Many of the problems are from the International Mathematical Olympiad (IMO) competition, the most challenging high school math competition in the world. Yet frontier models like Gemini 1.5 Pro can achieve 91.1 accuracy on this benchmark, which is far better than an average human. Another notable observation is the performance from open-source and small models – although there exist gaps between open-source and frontier models, the gap is not particularly large. For example, Intern LM2 Math [Ying et al., 2024] also achieves 68.5 on MATH, far better than the original GPT-3.5 performance (19.1, see Fu et al. [2023a]).

Fast Benchmark Climbing Speed Not only the models perform well on challenging benchmarks, their speed of improvement is remarkable. Figure 6.1 shows how fast models improve over the period from Feb 2023 to May 2024. When we studied finetuning FlanT5 in Dec 2022 (Chapter 4), our best GSM8K performance with an 11B model was 27, and we found it hard to imagine how can a 7B model perform better than 50 within 2 years. When we presented this work after 6 months at ICML 2023, there well still not a 7B model doing better than 50 on GSM8K. Yet during the second half of 2023, the 7B model performance reached about 90, far beyond our expectations. On MATH there was a similar story. In general, when complexity-based prompting (Chapter 3) and specializing T5 (Chapter 4) were proposed in late 2022, they achieved the SOTA performance on GSM8K and MATH respectively, yet now on Figure 6.1 they are at the bottom of the leaderboard.

Benchmark Saturation Except for GPQA, on MMLU, MATH and BBH, the leading closed-source models are all approaching 90+ accuracy, meaning that they are about to saturate these benchmarks. Open-source models, although slightly lagging behind, since they are improving at an accelerated speed, may approach close-source frontier models in the near future. Considering that the construction of these benchmarks was quite nontrivial and already involved expert-level human annotation, the construction of evaluation datasets that can distinguish frontier model capability will be a continuous research challenge.

Potential Benchmark Leakage Whether the model has already seen exactly the same, or similar problems during pretraining always cast a shadow on the high accuracy score, as criticized by Schaeffer [2023]. Here we distinguish three levels of generalization:

- **Memorization:** the model has seen the same question during training.
- **In-distribution Generalization:** the model has seen similar, but not identical questions.
- **Out-of-distribution Generalization:** the model has not seen even similar questions during training.

Existing literature like Bi et al. [2024], Huang et al. [2024] uses loss differences to test whether a model has seen similar or identical test questions and there is indeed strong evidence for test set leakage. Preventing contamination is a pressing effort that should be collectively enforced by the community.

Furthermore, the level of generalization and to what extent one should expect the model to generalize is a more delicate matter. Before the emergence of large language models, classical machine learning problems expect in-distribution generalization: one trains the model on a set of problems of a certain distribution, one expects the model to generalize on the same set of distributions. In the age of large-language models, due to the discovery of emergent ability and zero-shot generalization, people start to expect the model to generalize further than in-distribution [Gu et al., 2021], particularly in the instruction tuning literature, one tune the model on a certain set of instructions and expect the model to generalize to another set of instructions from another distribution that the model have not been tuned on [Wei et al., 2021b, Chung et al., 2024, Ouyang et al., 2022] (but might be seen in the pretraining stage). This leads to the question that when we say the model “has seen similar problem/ instruction”, how “similar” the problem should be? For many reasoning problems, since we not only expect the model to generate the final answer but also expect the model to generate the intermediate rationale, if a model has indeed seen the annotated rationale of a problem from the test set, but still come up with a brand new intermediate rationale (e.g., for math problems there could be multiple paths all leading to the correct answer [Wang et al., 2023b]), shall we say this model generalizes (since it comes up with its own solution) or not (since it has already seen a reference answer)? These questions still require detailed future study.

6.2 Future Directions

Text-Based Scaling is Likely Reaching a Limit The most recently released Llama3 [Meta, 2024] model uses 15T tokens for pretraining, which is so far the model that uses the

largest number of tokens and far beyond Chinchilla optimal (which suggests only 2-4 trillion tokens for a 70B model [Hoffmann et al., 2022]). Most of the data source is from CommonCrawl, Github, and academic publications like Arxiv, which are already used up by most of the data cleaning pipelines [Thoppilan et al., 2022, Young et al., 2024, Yang et al., 2023, Bai et al., 2023]. If one does not substantially relax the deduplication and filtering threshold (which may strongly influence the data quality), it is less likely that one can obtain another order of magnitude larger data. This is to say, one may still obtain incremental amounts of data from under-explored sources like library books, gather newly created data from the web, or construct synthetic data [Liu et al., 2024]. But these data are more likely to be within a smaller order of magnitude, say about 100B tokens, and it would be extremely hard to achieve a larger level of magnitude (say 50T) new text data. Eventually, to continue scaling up the pretraining tokens, one needs to gather data from other sources than text.

Multimodal Pretraining The most straightforward motivation for multimodal pretraining is that we are running out of text data, so we need to use multimodal, particularly video data to continue scaling up. Another important motivation is that the video perception capability opens a large array of possibilities from another dimension and enables applications (e.g., robot control [Zitkovich et al., 2023]) that text-only models can never do. If one can obtain, say another 100T tokens from videos on the internet, one may continue scaling up the model for at least two additional orders of magnitude. Currently, the unified video-language generative modeling research is still at a relatively early stage [Wadkar et al., 2024], and the most important direction is to develop a scalable architecture [Lu et al., 2024b, Yu et al., 2023a] that can simultaneously understand and generate video and text.

Online Deployment and Iterative Learning from X Feedback Although video data gives another order of magnitude larger amount of tokens, currently, it seems that video data cannot increase the reasoning capability [Lu et al., 2024a, Young et al., 2024, Lu et al., 2024b, Yu et al., 2023a]. Scaling up reinforcement learning from X feedback seems to be the most promising direction to continuing increase the model's reasoning capability, where X means human [Ouyang et al., 2022, Bai et al., 2022a], AI [Bai et al., 2022b, Fu et al., 2023b, Wang et al., 2023a], and environment feedback [Zhou et al., 2024]. Just like how AlphaGo Zero [Silver et al., 2017] achieves super-human performance on Go, self-play and interacting with the environment could be a direction for super-human generative models. Making the model online and iteratively learn [Guo et al., 2024, Bai et al., 2022a] from the feedback (instead of a single-round offline op-

timization) could potentially lead to continuously increased reasoning capability.

Putting Them Together: Towards a Continuously Improving Multimodal AI Agent

Eventually, we envisage a continuously improving multimodal AI agent. This agent might be pretrained on a large amount of multimodal data, whose scale might be orders of magnitude larger than the current largest multimodal model. This agent is also iteratively aligned with expert-level human feedback, as well as feedback from other strong AI agents and their environment. This agent is deployed to real-world scenarios and directly improves efficiency in areas like law, finance, software engineering, medicine, and so on. This agent is continuously improved on its own, by continuously and iteratively exploring its environment and exploiting its policies. This agent combines all the techniques that we have described so far and is constructed under collective innovation from the community, and under the oversight of strict safety regulations. Eventually, we hope to construct such a strong AI agent and maximally benefit human beings.

Chapter 7

Conclusions

In this thesis, we delve into the problem of improving language model’s complex reasoning. We discuss methods from three perspectives: in-context learning, finetuning, and learning from feedback, covering most of the important stages in post-training.

To improve reasoning by in-context learning, we use examples with complex reasoning paths (more steps) and significantly improve the model’s ability to solve complex problems. Complexity-based prompting has proven to be effective in math and logical reasoning tasks, achieving state-of-the-art results on benchmarks such as GSM8K and MATH. Furthermore, the idea of teaching the model with more complex problems consistently serves as a rule of thumb for training increasing stronger models. The level of difficulty increases from GSM8K (2022) to MATH (2023) to AIME (2024), and we would be excited to see how the complexity further increase to professional-level problems.

To improve reasoning by finetuning, we study the data format, finetuning procedure, and capability balancing problem. We show that smaller LLMs can be specialized for specific reasoning tasks like math by fine-tuning them on data generated by a larger, more capable model. This specialization significantly boosts performance on target tasks, but there is a trade-off with potential degradation in performance on generic tasks. Later work shows that by mixing and balancing instructions from different sources and capability directions, it would be possible to achieve targeted improvements without much sacrificing generic capabilities. Careful instruction data engineering, including format, quality, and diversity, is crucial for achieving good generalization and capability balancing.

To improve reasoning by learning from AI feedback, we show that LLMs can potentially learn and improve themselves autonomously in strategic reasoning games by

playing against each other and incorporating feedback from an AI critic. This self-play and AI feedback approach requires models to be both capable (understand rules) and aligned (respond to feedback). Strong, well-aligned models like GPT-4 and Claude-v1.3 have shown consistent improvement over multiple rounds.

We further review the current status of AI and language models. We see that frontier models are about to saturate many of the benchmarks, even these benchmarks are quite challenging for expert-level human. Open-source models, although lagging behind, are continuously improving and, in many cases, close to the frontier models. We also note that currently, text-based scaling is likely reaching a limit because it is hard to obtain another order of magnitude new data from the internet. Currently, the community's focus shifting towards discovering new dimensions of scaling, such as multimodal pre-training and scaling up inference-time compute. Deploying multimodal AI agents and letting them continuously evolve from human, AI, and environment feedback to solve real-world complex workflow would be important topics for future reasoning research.

Bibliography

- Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL <https://aclanthology.org/2023.emnlp-main.298>.
- Ekin Akyurek and Afra Feyza Akyurek. Notes on teaching gpt-3 adding numbers, 2022. URL https://lingo.csail.mit.edu//blog/arithmetic_gpt3.
- Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2022.
- Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1245. URL <https://aclanthology.org/N19-1245>.
- Jacob Andreas. Measuring compositionality in representation learning. In *International Conference on Learning Representations*, 2018.
- Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024. URL https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf.

Amanda Askeell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=4WnqRR915j>.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askeell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askeell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Paul Barham, Aakanksha Chowdhery, Jeff Dean, Sanjay Ghemawat, Steven Hand, Daniel Hurt, Michael Isard, Hyeontaek Lim, Ruoming Pang, Sudip Roy, et al. Pathways: Asynchronous distributed dataflow for ml. *Proceedings of Machine Learning and Systems*, 4:430–449, 2022.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013*

Conference on Empirical Methods in Natural Language Processing, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL <https://aclanthology.org/D13-1160>.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

Samuel R Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen, Craig Pettit, Scott Heiner, Kamile Lukosuite, Amanda Askell, Andy Jones, Anna Chen, et al. Measuring progress on scalable oversight for large language models. *arXiv preprint arXiv:2211.03540*, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020a. URL <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla

- Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020b.
- Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=qV83K9d5WB>.
- Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.
- Kushal Chawla, Jaysa Ramirez, Rene Clever, Gale Lucas, Jonathan May, and Jonathan Gratch. CaSiNo: A corpus of campsite negotiation dialogues for automatic negotiation systems. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3167–3185, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.254. URL <https://aclanthology.org/2021.naacl-main.254>.
- Angela Chen, Luke Yeager, and Janne Blomqvist. Understanding selene: A modular nvidia supercomputer. Technical report, Nvidia, 2021a. URL <https://www.nvidia.com/en-us/on-demand/session/gtcspring21-s31700/>.
- Angelica Chen, Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Samuel R. Bowman, Kyunghyun Cho, and Ethan Perez. Learning from natural language feedback. *Transactions on Machine Learning Research*, 2024a. ISSN 2835-8856. URL <https://openreview.net/forum?id=xo3hI5MwvU>.
- Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint arXiv:2310.05029*, 2023a.
- Jiangjie Chen, Siyu Yuan, Rong Ye, Bodhisattwa Prasad Majumder, and Kyle Richardson. Put your money where your mouth is: Evaluating strategic planning and execution of llm agents in an auction arena. *arXiv preprint arXiv:2310.05746*, 2023b.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021b.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *The Twelfth International Conference on Learning Representations*, 2024b. URL <https://openreview.net/forum?id=KuPixIqPiq>.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. Learning an Executable Neural Semantic Parser. *Computational Linguistics*, 45(1):59–94, 03 2019. ISSN 0891-2017. doi: 10.1162/coli_a_00342. URL https://doi.org/10.1162/coli_a_00342.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayanan Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1), mar 2024. ISSN 1532-4435.

Hyung Won Chung. Large language models (in 2023), 2023. URL <https://www.youtube.com/watch?v=dbo3kNKPauA>.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25 (70):1–53, 2024.

- Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Cohere. Cohere command models. *Cohere website*, 2023. URL <https://docs.cohere.com/docs/models>.
- Together Computer. Redpajama: an open dataset for training large language models, 2023. URL <https://github.com/togethercomputer/RedPajama-Data>.
- Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=3Pf3Wg6o-A4>.
- Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, 2023.
- DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen,

S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, T. Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Liu, Xin Xie, Xingkai Yu, Xinnan Song, Xinyi Zhou, Xinyu Yang, Xuan Lu, Xuecheng Su, Y. Wu, Y. K. Li, Y. X. Wei, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Zheng, Yichao Zhang, Yiliang Xiong, Yilong Zhao, Ying He, Ying Tang, Yishi Piao, Yixin Dong, Yixuan Tan, Yiyuan Liu, Yongji Wang, Yongqiang Guo, Yuchen Zhu, Yudian Wang, Yuheng Zou, Yukun Zha, Yunxian Ma, Yuting Yan, Yuxiang You, Yuxuan Liu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhewen Hao, Zhihong Shao, Zhiniu Wen, Zhipeng Xu, Zhongyu Zhang, Zhuoshu Li, Zihan Wang, Zihui Gu, Zilin Li, and Ziwei Xie. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Nan Duan, Duyu Tang, and Ming Zhou. Machine reasoning: Technology, dilemma and future. In Aline Villavicencio and Benjamin Van Durme, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 1–6, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-tutorials.1. URL <https://aclanthology.org/2020.emnlp-tutorials.1>.

- Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014. URL <https://api.semanticscholar.org/CorpusID:207214527>.
- Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-SQL evaluation methodology. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1033. URL <https://aclanthology.org/P18-1033>.
- Yao Fu, Hao Peng, and Tushar Khot. How does gpt obtain its ability? tracing emergent abilities of language models to their sources. *Yao Fu’s Notion*, Dec 2022a.
- Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*, 2022b.
- Yao Fu, Litu Ou, Mingyu Chen, and Yuhao Wan. Measuring llms’ reasoning performance. *Github*, 2023a. URL <https://github.com/FranxYao/chain-of-thought-hub>.
- Yao Fu, Hao Peng, Tushar Khot, and Mirella Lapata. Improving language model negotiation with self-play and in-context learning from ai feedback. *arXiv preprint arXiv:2305.10142*, 2023b.
- Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10421–10430. PMLR, 23–29 Jul 2023c. URL <https://proceedings.mlr.press/v202/fu23d.html>.
- Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023.

- 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. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.
- Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.
- Google. Palm 2 technical report. *ArXiv*, 2023.
- Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A*, 478(2266):20210068, 2022.
- Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488, 2021.
- Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *Sort*, 2(4):0–6.
- Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.
- Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.830. URL <https://aclanthology.org/2023.acl-long.830>.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan

- Clark, et al. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pages 30016–30030, 2022.
- Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.67. URL <https://aclanthology.org/2023.emnlp-main.67>.
- Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.67. URL <https://aclanthology.org/2023.findings-acl.67>.
- Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32, 2019.
- Yuzhen Huang, Jinghan Zhang, Zifei Shan, and Junxian He. Compression represents intelligence linearly. *arXiv preprint arXiv:2404.09937*, 2024.
- Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.
- Mohit Iyyer, Jordan L. Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé. A neural network for factoid question answering over paragraphs. In *Conference on Empirical Methods in Natural Language Processing*, 2014. URL <https://api.semanticscholar.org/CorpusID:216034672>.
- Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Deendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guil-

- laume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. VIMA: General robot manipulation with multimodal prompts. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022. URL <https://openreview.net/forum?id=oU2DzdTI94>.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.
- PN Johnson-Laird. 15 a taxonomy of thinking. *The psychology of human thought*, page 429, 1988.
- Nitish Joshi, Hanlin Zhang, Koushik Kalyanaraman, Zhiting Hu, Kumar Chellapilla, He He, and Li Erran Li. Improving multi-hop reasoning in LLMs by learning from rich human feedback. In *Neuro-Symbolic Learning and Reasoning in the era of Large Language Models*, 2023. URL <https://openreview.net/forum?id=wxfqhp9bNR>.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Text modular networks: Learning to decompose tasks in the language of existing models. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1264–1279, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.99. URL <https://aclanthology.org/2021.naacl-main.99>.
- Tushar Khot, Kyle Richardson, Daniel Khashabi, and Ashish Sabharwal. Hey AI, can you solve complex tasks by talking to agents? In Smaranda Muresan, Preslav

- Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1808–1823, Dublin, Ireland, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.142. URL <https://aclanthology.org/2022.findings-acl.142>.
- Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*, 2022b.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.
- AI21 Labs. Announcing jurassic-2 and task-specific apis. *AI21 Blog*, 2023. URL <https://www.ai21.com/blog/introducing-j2>.
- Yuxuan Lai, Chen Zhang, Yansong Feng, Quzhe Huang, and Dongyan Zhao. Why machine reading comprehension models learn shortcuts? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 989–1002, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.85. URL <https://aclanthology.org/2021.findings-acl.85>.
- Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR, 2018.
- Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.
- Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*, 2022.
- Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328, 2022.

- Jacqueline P. Leighton and Robert J. Sternberg. *The Nature of Reasoning*. Cambridge University Press, 2003. doi: 10.1017/CBO9780511818714.
- Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? end-to-end learning of negotiation dialogues. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1259. URL <https://aclanthology.org/D17-1259>.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.
- Chengpeng Li, Zheng Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. Query and response augmentation cannot help out-of-domain math reasoning generalization. *arXiv preprint arXiv:2310.05506*, 2023a.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL <https://openreview.net/forum?id=3IyL2XWdkG>.
- Shiyang Li, Jianshu Chen, yelong shen, Zhiyu Chen, Xinlu Zhang, Zekun Li, Hong Wang, Jing Qian, Baolin Peng, Yi Mao, Wenhua Chen, and Xifeng Yan. Explanations from large language models make small reasoners better. In *2nd Workshop on Sustainable AI*, 2024. URL <https://openreview.net/forum?id=rH8ZUcfL9r>.
- Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.
- Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Making language models better reasoners with step-aware verifier. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of*

the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5315–5333, Toronto, Canada, July 2023c. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.291. URL <https://aclanthology.org/2023.acl-long.291>.

Percy Liang. Learning executable semantic parsers for natural language understanding. *Communications of the ACM*, 59(9):68–76, 2016.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL <https://openreview.net/forum?id=i04LZibEqW>. Featured Certification, Expert Certification.

Austen Liao, Nicholas Tomlin, and Dan Klein. Efficacy of language model self-play in non-zero-sum games. *arXiv preprint arXiv:2406.18872*, 2024.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Hao Liu, Carmelo Sferrazza, and Pieter Abbeel. Chain of hindsight aligns language models with feedback. In *The Twelfth International Conference on Learning Representations*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland

and Online, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.deelio-1.10. URL <https://aclanthology.org/2022.deelio-1.10>.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI'20*, 2021. ISBN 9780999241165.

Linqing Liu, Patrick Lewis, Sebastian Riedel, and Pontus Stenetorp. Challenges in generalization in open domain question answering. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2014–2029, Seattle, United States, July 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.155. URL <https://aclanthology.org/2022.findings-naacl.155>.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9), jan 2023. ISSN 0360-0300. doi: 10.1145/3560815. URL <https://doi.org/10.1145/3560815>.

Ruibo Liu, Jason Wei, Shixiang Shane Gu, Te-Yen Wu, Soroush Vosoughi, Claire Cui, Denny Zhou, and Andrew M Dai. Mind’s eye: Grounded language model reasoning through simulation. In *The Eleventh International Conference on Learning Representations*, 2022c.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, et al. Best practices and lessons learned on synthetic data for language models. *arXiv preprint arXiv:2404.07503*, 2024.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pages 22631–22648. PMLR, 2023.

Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Yaofeng Sun, et al. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024a.

- Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26439–26455, 2024b.
- Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, 2022.
- Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *2nd Workshop on Language and Robot Learning: Language as Grounding*, 2023. URL <https://openreview.net/forum?id=WbFBZ3bQUs>.
- Aman Madaan, Niket Tandon, Dheeraj Rajagopal, Yiming Yang, Peter Clark, Keisuke Sakaguchi, and Ed Hovy. Improving neural model performance through natural language feedback on their explanations. *arXiv preprint arXiv:2104.08765*, 2021.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=S37hOerQLB>.
- Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1773–1781, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.151. URL <https://aclanthology.org/2023.acl-short.151>.
- Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irv-

- ing, et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.
- Meta. Llama 3 release blog, 2024. URL <https://llama.meta.com/llama3/>.
- FAIR Meta, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022.
- Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.201. URL <https://aclanthology.org/2022.naacl-main.201>.
- Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. In *60th Annual Meeting of the Association for Computational Linguistics, ACL 2022*, pages 3470–3487. Association for Computational Linguistics (ACL), 2022.
- Mistral. Mixtral 8*22b release blog, 2024. URL <https://mistral.ai/news/mixtral-8x22b/>.
- Arindam Mitra, Luciano Del Corro, Guoqing Zheng, Shweti Mahajan, Dany Rouhana, Andres Cudas, Yadong Lu, Wei-ge Chen, Olga Vrousgos, Corby Rosset, et al. Agentinstruct: Toward generative teaching with agentic flows. *arXiv preprint arXiv:2407.03502*, 2024.
- Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1176. URL <https://aclanthology.org/P18-1176>.
- Gabriel Mukobi, Hannah Erlebach, Niklas Lauffer, Lewis Hammond, Alan Chan, and Jesse Clifton. Welfare diplomacy: Benchmarking language model cooperation. *arXiv preprint arXiv:2310.08901*, 2023.

- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. *Webgpt: Browser-assisted question-answering with human feedback*. *arXiv preprint arXiv:2112.09332*, 2021.
- Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–53, 1970.
- Mike Oaksford and Nick Chater. *Bayesian rationality: The probabilistic approach to human reasoning*. Oxford University Press, 2007.
- OpenAI. Introducing chatgpt. *OpenAI Blog*, 2022. URL <https://openai.com/blog/chatgpt>.
- OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. URL <https://api.semanticscholar.org/CorpusID:257532815>.
- OpenDevin Team. OpenDevin: An Open Platform for AI Software Developers as Generalist Agents. <https://github.com/OpenDevin/OpenDevin>, 2024. Accessed: ENTER THE DATE YOU ACCESSED THE PROJECT.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.
- Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.
- Pruthvi Patel, Swaroop Mishra, Mihir Parmar, and Chitta Baral. Is a question decomposition unit all we need? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4553–4569, 2022.
- Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon LLM: Outperforming curated corpora

with web data only. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2023*. URL <https://openreview.net/forum?id=kM5eGcdCzq>.

Ethan Perez, Sam Ringer, Kamilė Lukošiuūtė, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, et al. Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*, 2022.

PromptBase. Promptbase. <https://promptbase.com>.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. Reasoning with language model prompting: A survey. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.294. URL <https://aclanthology.org/2023.acl-long.294>.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems, 2023*. URL <https://openreview.net/forum?id=HPuSIXJaa9>.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

- Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. 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 3982–3992, 2019.
- Kyle Richardson and Ashish Sabharwal. Pushing the limits of rule reasoning in transformers through natural language satisfiability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11209–11219, 2022.
- Frieda Rong. Extrapolating to unnatural language processing with gpt-3’s in-context learning: The good, the bad, and the mysterious. <http://ai.stanford.edu/blog/in-context-learning/>.
- Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1202. URL <https://aclanthology.org/D15-1202>.
- Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.
- Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.191. URL <https://aclanthology.org/2022.naacl-main.191>.
- Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, et al. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2021.
- Rylan Schaeffer. Pretraining on the test set is all you need. *arXiv preprint arXiv:2309.08632*, 2023.

- Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=Yacmpz84TH>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- Pavel Senin. Dynamic time warping algorithm review. 2008.
- Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*, 2017. URL <https://openreview.net/forum?id=HJ0UKP9ge>.
- Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. ZeroSCROLLS: A zero-shot benchmark for long text understanding. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7977–7989, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.536. URL <https://aclanthology.org/2023.findings-emnlp.536>.
- Christopher J Shallue, Jaehoon Lee, Joseph Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *Journal of Machine Learning Research*, 20(112):1–49, 2019.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- Sheng Shen, Le Hou, Yanqi Zhou, Nan Du, Shayne Longpre, Jason Wei, Hyung Won Chung, Barret Zoph, William Fedus, Xinyun Chen, Tu Vu, Yuexin Wu, Wuyang Chen, Albert Webson, Yunxuan Li, Vincent Y Zhao, Hongkun Yu, Kurt Keutzer, Trevor Darrell, and Denny Zhou. Mixture-of-experts meets instruction tuning: A winning combination for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=6mLjDwYte5>.

- Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.346. URL <https://aclanthology.org/2020.emnlp-main.346>.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.
- Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling reasoning capabilities into smaller language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.441. URL <https://aclanthology.org/2023.findings-acl.441>.
- Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. Prompting gpt-3 to be reliable. In *The Eleventh International Conference on Learning Representations*, 2022.
- David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
- David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.
- Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, et al. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*, 2023.

- Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhunoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.
- Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Khyathi Chandu, Jennifer Dumas, Li Lucy, Xinxu Lyu, Ian Magnusson, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Pete Walsh, Hannaneh Hajishirzi, Noah A. Smith, Luke Zettlemoyer, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: An Open Corpus of 3 Trillion Tokens for Language Model Pretraining Research. Technical report, Allen Institute for AI, 2023. Released under ImpACT License as Medium Risk artifact, <https://github.com/allenai/dolma>.
- Keith E. Stanovich and Richard F. West. Individual differences in reasoning: Implications for the rationality debate? *Behavioral and Brain Sciences*, 23(5):645–665, 2000. doi: 10.1017/S0140525X00003435.
- Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*, 2022.
- Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. What makes reading comprehension questions easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1453. URL <https://aclanthology.org/D18-1453>.
- Tian-Xiang Sun, Xiang-Yang Liu, Xi-Peng Qiu, and Xuan-Jing Huang. Paradigm shift in natural language processing. *Machine Intelligence Research*, 19(3):169–183, 2022.

- Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. Principle-driven self-alignment of language models from scratch with minimal human supervision. *Advances in Neural Information Processing Systems*, 36, 2024.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL <https://aclanthology.org/2023.findings-acl.824>.
- Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=S1gUsoR9YX>.
- Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Dara Bahri, Tal Schuster, Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. UL2: Unifying language learning paradigms. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=6ruVLB727MC>.
- Parth Thakkar. Copilot explorer. *thakkarparth007.github.io*, 2022. URL <https://thakkarparth007.github.io/copilot-explorer/posts/copilot-internals.html>.
- Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Teaching broad reasoning skills for multi-step QA by generating hard contexts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6541–6566, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.439. URL <https://aclanthology.org/2022.emnlp-main.439>.
- Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.
- Shakti N Wadekar, Abhishek Chaurasia, Aman Chadha, and Eugenio Culurciello. The evolution of multimodal model architectures. *arXiv preprint arXiv:2405.17927*, 2024.
- Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. 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 2153–2162, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1221. URL <https://aclanthology.org/D19-1221>.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL <https://openreview.net/forum?id=ehfRiF0R3a>.

- Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Y Wu, and Zhifang Sui. Math-shepherd: A label-free step-by-step verifier for llms in mathematical reasoning. *arXiv preprint arXiv:2312.08935*, 2023a.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Rationale-augmented ensembles in language models. *arXiv preprint arXiv:2207.00747*, 2022a.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL <https://openreview.net/forum?id=1PL1NIMMrw>.
- Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, 2022b.
- Colin Wei, Sang Michael Xie, and Tengyu Ma. Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning. *Advances in Neural Information Processing Systems*, 34:16158–16170, 2021a.
- Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2021b.
- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022a. ISSN 2835-8856. URL <https://openreview.net/forum?id=yzkSU5zdwD>. Survey Certification.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022b.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. Skywork: A more open bilingual foundation model, 2023.

Lilian Weng. Llm-powered autonomous agents. *lilianweng.github.io*, Jun 2023. URL <https://lilianweng.github.io/posts/2023-06-23-agent/>.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2021.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, et al. Agentgym: Evolving large language model-based agents across diverse environments. *arXiv preprint arXiv:2406.04151*, 2024.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=RdJVfCHjUMI>.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. In *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=c1AKcA6ry1>.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

- Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. Search-in-the-chain: Interactively enhancing large language models with search for knowledge-intensive tasks. In *The Web Conference 2024*, 2024. URL <https://openreview.net/forum?id=tr0TcqitMH>.
- Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. Exploring large language models for communication games: An empirical study on werewolf. *arXiv preprint arXiv:2309.04658*, 2023.
- Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL <https://aclanthology.org/D18-1259>.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2022b.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=5Xc1ecx01h>.
- Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. StructVAE: Tree-structured latent variable models for semi-supervised semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 754–765, Melbourne, Australia, July 2018.

Association for Computational Linguistics. doi: 10.18653/v1/P18-1070. URL <https://aclanthology.org/P18-1070>.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, Yudong Wang, Zijian Wu, Shuaibin Li, Fengzhe Zhou, Hongwei Liu, Songyang Zhang, Wenwei Zhang, Hang Yan, Xipeng Qiu, Jiayu Wang, Kai Chen, and Dahua Lin. Internlm-math: Open math large language models toward verifiable reasoning, 2024.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

Lili Yu, Bowen Shi, Ramakanth Pasunuru, Benjamin Muller, Olga Golovneva, Tianlu Wang, Arun Babu, Binh Tang, Brian Karrer, Shelly Sheynin, et al. Scaling autoregressive multi-modal models: Pretraining and instruction tuning. *arXiv preprint arXiv:2309.02591*, 2(3), 2023a.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=N8N0hgNDRt>.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Kaixin Ma, Hongwei Wang, and Dong Yu. Chain-of-note: Enhancing robustness in retrieval-augmented language models. *arXiv preprint arXiv:2311.09210*, 2023b.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhua Chen. MAMmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024a. URL <https://openreview.net/forum?id=yLClGs770I>.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhua Chen. Mammoth2: Scaling instructions from the web. *arXiv preprint arXiv:2405.03548*, 2024b.

- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hel-laSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL <https://www.aclweb.org/anthology/P19-1472>.
- Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.
- Wenting Zhao, Mor Geva, Bill Yuchen Lin, Michihiro Yasunaga, Aman Madaan, and Tao Yu. Complex reasoning in natural language. In Yun-Nung (Vivian) Chen, Margot Margot, and Siva Reddy, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 11–20, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-tutorials.2. URL <https://aclanthology.org/2023.acl-tutorials.2>.
- Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/zhao21c.html>.
- Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*, 2023.
- Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, and Hongsheng Li. Solving challenging math word problems using GPT-4 code interpreter with code-based self-verification. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=c8McWs4Av0>.
- Ben Zhou, Kyle Richardson, Xiaodong Yu, and Dan Roth. Learning to decompose: Hypothetical question decomposition based on comparable texts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2223–2235, Abu

- Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.142. URL <https://aclanthology.org/2022.emnlp-main.142>.
- Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=KBMOkMx2he>.
- Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 06–09 Nov 2023. URL <https://proceedings.mlr.press/v229/zitkovich23a.html>.
- Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.