Unveiling the Impact of Coding Data Instruction Fine-Tuning on Large Language Models Reasoning

Xinlu Zhang[1*], **Zhiyu Zoey Chen**[2], **Xi Ye**[3], **Xianjun Yang**[1], **Lichang Chen**[4], **William Yang Wang**[1], **Linda Ruth Petzold** [1]

[1]University of California, Santa Barbara
[2]The University of Texas at Dallas
[3]The University of Texas at Austin
[4]University of Maryland, College Park

Abstract

Instruction Fine-Tuning (IFT) significantly enhances the zero-shot capabilities of pretrained Large Language Models (LLMs). While coding data is known to boost LLM reasoning abilities during pretraining, its role in activating internal reasoning capacities during IFT remains understudied. This paper investigates a key question: *How does coding data impact LLMs' reasoning capacities during IFT stage?* To explore this, we thoroughly examine the impact of coding data across different coding data proportions, model families, sizes, and reasoning domains, from various perspectives. Specifically, we create three IFT datasets with increasing coding data proportions, fine-tune six LLM backbones across different families and scales on these datasets, evaluate the tuned models' performance across twelve tasks in three reasoning domains, and analyze the outcomes from three broad-to-granular perspectives: overall, domain-level, and task-specific. Our holistic analysis provides valuable insights into each perspective. First, coding data tuning enhances the overall reasoning capabilities of LLMs across different model families and scales. Moreover, while the impact of coding data varies by domain, it shows consistent trends within each domain across different model families and scales. Additionally, coding data generally provides comparable task-specific benefits across model families, with optimal proportions in IFT datasets being task-dependent.

Extended version — https://arxiv.org/abs/2405.20535

1 Introduction

Large Language Models (LLMs) have significantly advanced in task generalization by training on diverse text data sources (Touvron et al. 2023a,b; Brown et al. 2020; Jiang et al. 2023) and instruction-finetuning (IFT) further elicits their intrinsic abilities in a zero-shot manner (Ouyang et al. 2022; Longpre et al. 2023; Wei et al. 2022a; OpenAI 2022, 2023; Anthropic 2023). Although previous studies (Peng et al. 2023; Taori et al. 2023; Xu et al. 2023) have improved IFT dataset diversity to better align LLMs with human needs, the impact of specific data types during IFT remains underexplored.

Coding data, with its logical consistency and reduced ambiguity compared to natural text, has been empirically shown to enhance LLM reasoning capabilities during pretraining (Liang et al. 2023; Fu and Khot 2022; Ma et al. 2023; Guo et al. 2024). This enables LLMs to acquire advanced intrinsic knowledge, supporting complex reasoning in real-world applications like text summarization (Yang et al. 2023), numerical problem solving (Luo et al. 2023a; Yue et al. 2023), and knowledge-intensive tasks (Chen et al. 2024). However, at the IFT stage, research primarily shows that coding data tuning improves coding-related in-domain performance (Yuan et al. 2023; Luo et al. 2023b; Ma et al. 2023). The impact on **out-of-domain general reasoning capabilities** remains underexplored due to complex variations in coding data proportions, model backbones, and reasoning task types (Wei et al. 2022c; Ma et al. 2023; Liang et al. 2023). Therefore, we raise a natural question: *How does coding data impact LLMs' reasoning capacities during the IFT stage?*

To thoroughly answer this question, we propose an analysis pipeline to investigate how coding data affects LLMs' reasoning capacities during IFT, considering coding data proportions, model families, scales, and reasoning domains. We create IFT datasets from ShareGPT (Sharegpt 2023) using ChatGPT (OpenAI 2022) to classify instances as either coding or general text, forming code-centric and general textual datasets. We then generate three IFT datasets with coding data proportions of 0%, 50%, and 100%, maintaining consistent dataset sizes. Six base models of varying families and scales —Llama-1 (Touvron et al. 2023a), Llama-2 (Touvron et al. 2023b), Llama-3 (AI@Meta 2024), Mistral (Jiang et al. 2023), Qwen-1.5 (Bai et al. 2023), and Gemma (Team et al. 2024)—are fine-tuned on these datasets. We evaluate the tuned models on symbolic, logical, and arithmetic reasoning domains. Finally, we analyze coding data impact from three perspectives: overall effectiveness, domain-level influence, and task-specific performance, as shown in Figure 1.

To our best knowledge, this is the first work to thoroughly analyze how coding data affects LLMs' reasoning capacities during IFT, across different coding data proportions, model families, scales, and reasoning types. We gain significant insights and summarize the main findings for each perspective.

Overall effectiveness. As the proportion of coding data for tuning increases, we observe consistent and gradual performance enhancements across different model families and scales. However, improvements vary among model back-

*Corresponding Author: xinluzhang@ucsb.edu

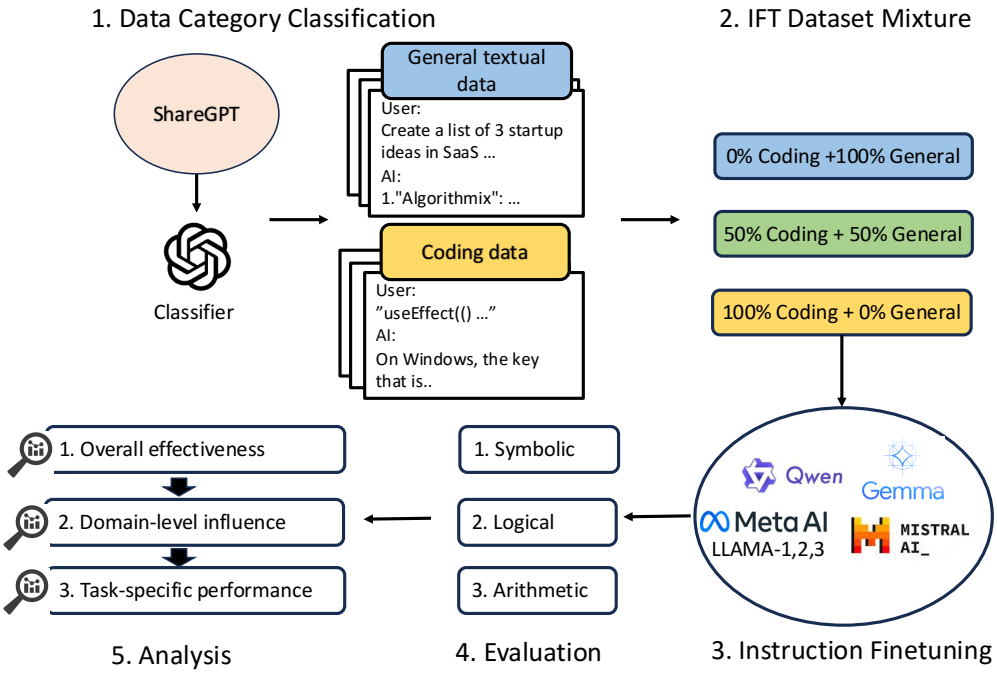

Figure 1: **Pipeline Overview.** The process utilizes ShareGPT data as a starting point. **1. Data Category Classification:** Using ChatGPT to classify code instances within ShareGPT to obtain a code-centric IFT dataset. **2. IFT Data Mixture:** Constructing three IFT mixture datasets with increasing proportions of coding data. **3. Instruction Finetuning:** Fine-tuning LLMs from six families across various scales with these three IFT datasets, respectively. **4. Evaluation:** Evaluating the fine-tuned models' reasoning capacities across three domains. **5. Analysis:** Analyzing from three broad-to-granular perspectives.

bones. Compared to tuning on the pure natural text dataset, the greatest overall improvement is achieved on Mistral with a 10.3 percentage points gain, while a more modest 1.5 percentage points gain obtained on Llama-3. Analysis of the models' responses indicates that coding data tuning enhances overall reasoning capacities for problem-solving, extending beyond in-domain programming skills.

Domain-level influence. Coding data tuning elicits different reasoning abilities in varied ways. There is marked improvement in the symbolic domain, which includes foundational reasoning skills. However, in arithmetic reasoning, which involves real-world math problems, performance gaps appear compared to models tuned on general textual datasets, addressing more diverse human needs. Additionally, we observe consistent performance trends across various model backbones and sizes within each reasoning domain, indicating the potential transferability of coding data effects during the IFT stage. Further analysis shows that models tuned with coding data can more adeptly apply appropriate skills for solving questions based on different domain properties.

Task-specific performance. Coding data typically yields comparable task-specific benefits across different model families, with a similar number of tasks showing improvement in two out of three reasoning domains. However, obtaining optimizing strategies for mixing coding and natural textual data presents a challenge. While the majority of optimal coding data proportions for improving task performance remain consistent across model families, there is no single coding data proportion setting that consistently enhances task-specific reasoning abilities better than another.

2 Related Work

IFT. IFT has proven effective in enhancing pretrained LLMs for zero-shot tasks (Ouyang et al. 2022; Longpre et al. 2023). Wei et al. (2022a); Chung et al. (2022) used IFT datasets from NLP benchmarks, improving generalization but often not fully aligning with real-world user intentions due to simpler instructions. In contrast, Ouyang et al. (2022) developed InstructGPT by tuning on a diverse dataset of real-world instructions and responses, better meeting user needs. Open-source models (Taori et al. 2023; Xu et al. 2023; Chiang et al. 2023; Peng et al. 2023; Zhang et al. 2024) fine-tuned on diverse IFT datasets from strong teacher models have shown that such diversity enhances alignment with complex user intents (Chen et al. 2023). However, the impact of different IFT data types on LLM effectiveness remains underexplored. This work studies the effect of coding data, known for logical clarity and structure, during IFT.

Reasoning in LLMs. Reasoning involves logically analyzing a subject, using evidence and prior knowledge to reach conclusions (Wason and Johnson-Laird 1972; Wason 1968). It has been seen as one of LLMs' emergent behaviors, shown as models are large enough (Wei et al. 2022b,c). Although improving models' reasoning capacities shows promising results in different applications (Li et al. 2022; Zhang et al. 2023), comprehensively evaluating these capacities remains

challenging due to their complex nature, requiring different fine-grained abilities within different subdomains (Wei et al. 2022c; Ma et al. 2023; Liang et al. 2023; Qiu et al. 2023). We aim to thoroughly evaluate models across the subdomains of symbolic, logical, and arithmetic reasoning to deepen understanding of LLMs' reasoning capabilities at the IFT stage.

Code in LLMs. Integrating code into LLMs enhances their performance in programming, complex reasoning, and structural knowledge capture (Ma et al. 2023; Liang et al. 2023; Fu and Khot 2022; Wang, Li, and Ji 2023; Madaan et al. 2022). Most investigations focus on the pretraining stage. Ma et al. (2023) shows that LLMs pretrained with coding data outperform those trained with only natural language in both code-related and general tasks. Liang et al. (2023) demonstrates that Codex excels in complex mathematical reasoning, and Madaan et al. (2022) highlights coding data's role in improving structural reasoning. During the IFT stage, Ma et al. (2023) reveals that coding data enhances in-domain abilities. Conversely, we study how coding data elicits out-of-domain reasoning capacities in pretrained LLMs.

3 Experimental Setting

IFT Data Construction

We use ShareGPT (Sharegpt 2023) as our data source. After deduplication and extraction of the initial round of Human-AI conversations, we obtain a dataset of 45,742 instances. Using GPT-3.5-turbo (OpenAI 2022), we categorize conversations into three groups: `Code`, `Math`, and `Others`. Conversations involving coding data are classified as `Code`, those related to mathematical concepts and problems as `Math`, and all other general natural language texts as `Others`. The categorization results in 10,196 `Code`, 1,481 `Math`, and 34,065 `Others` instances. We exclude the `Math` category to prevent its potential influence on the model's reasoning capabilities during tuning. For further experimentation, we select a random subset from the `Others` category, termed `General`, equal in size to the `Code` category, for fair analysis. We establish a `Half-half` setting by mixing equal portions of data from the `Code` and `General` categories. This setup produces three equal-size IFT datasets: `General`, `Half-half`, and `Code`, containing 0%, 50%, and 100% coding data, respectively. The detailed categorization prompts for ShareGPT and coding data, and the corresponding code category diversity analysis are in the Appendix.

Task Description

To thoroughly assess the models' reasoning capabilities, we evaluate them on twelve generative tasks, across three reasoning types: symbolic, logical, and arithmetic. **Symbolic**: We focus on four tasks (Wei et al. 2022c): (1) First Letter Concatenation, (2) Last Letter Concatenation, (3) Reverse List, and (4) Coin Flip. **Logical**: We utilize four tasks, requiring strong logical ability: (1) Cluttr (Sinha et al. 2019), (2) List Functions (Rule 2020), (3) Babi-Induction and (4) Babi-Deduction (Weston et al. 2015). **Arithmetic**: Four arithmetic benchmarks are involved to evaluate the mathematics world problem-solving ability (1) GSM8K (Cobbe et al. 2021), (2) SVAMP (Patel, Bhattamishra, and Goyal 2021), (3) ASDiv (Miao, Liang, and Su 2020), and (4) MAWPS (Koncel-Kedziorski et al. 2016). For symbolic reasoning, we generate synthetic datasets following Fortes (2023) and balance the representation of difficulty levels for each task. For example, we generate 500 instances for names containing 2 to 4 words in letter concatenation tasks. For other tasks, we evaluate the models with the test sets for each task when publicly available. Otherwise, the development sets are used. Details on data statistics, the reasoning domain selection and synthetic datasets generation are in the Appendix.

Instruction Fine-tuned LLMs

To systematically assess the coding data impact, we conduct experiments with six distinct LLM families: Llama-1 (Touvron et al. 2023a), Llama-2(Touvron et al. 2023b), Llama-3 (AI@Meta 2024), Mistral-v0.1 (Jiang et al. 2023), Qwen-1.5 (Bai et al. 2023), and Gemma (Team et al. 2024). Each model is fine-tuned on the three uniquely composed IFT datasets—`General`, `Half-half`, and `Code`, respectively. Each fine-tuned model is evaluated on the twelve reasoning tasks in three reasoning domains. The training prompt and hyperparameter settings are available in the Appendix.

Evaluation Setup

Our evaluation operates in a *zero-shot* setting, where models are prompted to generate responses to corresponding questions without additional context, minimizing external influences. We standardize experimental conditions by limiting the maximum token length to 1024 and employing greedy decoding for all model outputs. Given the variability in output styles, we utilize GPT-3.5-turbo (OpenAI 2022) as an extractor to parse predictions from the generated text. These predictions are subsequently compared to the ground truth using accuracy (%) for evaluation. The generation prompt for datasets and answer extraction prompt are in the Appendix.

4 Experimental Results

Results on Overall Performance

We first compare the average results of 12 reasoning tasks between models tuned with different coding data proportions across 6 LLM families. Results are shown in Table 1.

Coding data successfully elicits LLM reasoning capacities in IFT. Overall, we observed a gradual improvement in average accuracy as the proportion of coding data increased during the IFT stage. These consistent gains across different model families clearly demonstrate the benefits of the specialized knowledge that coding data provides, effectively enhancing models' reasoning capabilities. Notably, for using Mistral-7B-v0.1 as the base model, tuning with `Code`, achieves a significant absolute performance gain of 10.3 compared to tuning with `General`. This underscores that datasets rich in code are crucial for eliciting the advanced reasoning abilities of LLMs.

The improvements brought by coding data are divergent across model families. Although different model families show positive effects from tuning with coding data during IFT, the improvements vary across model families. For example, the Mistral-7B-v0.1 and Gemma-7B exhibit substantial

Coding data prop.(%)	Llama-1-7B	Llama-2-7B	Llama-3-8B	Mistral-7B-v0.1	Qwen-1.5-7B	Gemma-7B
`General` (0%)	23.2	26.7	41.4	42.0	42.2	22.3
`Half-half` (50%)	25.0	29.2	**42.9**	43.9	44.9	23.7
`Code` (100%)	**27.9**	**30.6**	42.6	**52.3**	**47.4**	**30.0**
Δ	+4.6	+3.9	+1.5	+10.3	+ 5.5	+ 7.7
η	+20.0%	+14.5%	+3.6%	+24.6%	+13.0%	+34.7%

Table 1: **Overall reasoning comparison of models tuned on `General`, `Half-half`, and `Code` with increasing coding data proportion, across different families.** Results show average scores (%) across 12 reasoning tasks. The **Best** setting per model family is in bold. Δ and η indicate absolute and relative gains, respectively, between the best-performing coding data setting and `General`.

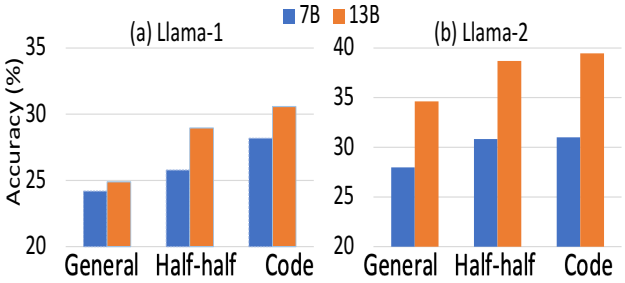

Figure 2: **Overall comparison across different scales.** Results show the overall performance of models using Llama-1 and Llama-2 with 7B and 13B parameters as backbones.

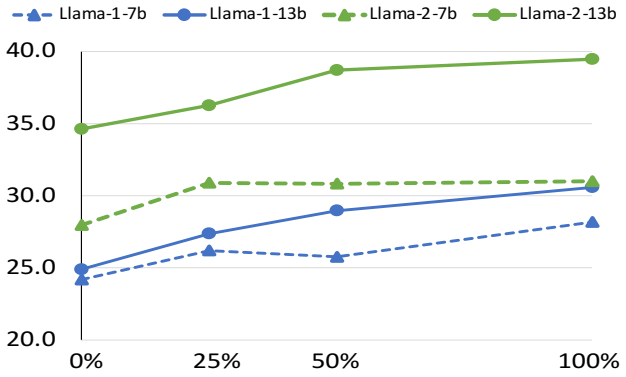

Figure 3: Overall performance comparison across different coding data percentages.

benefits, achieving 24.6% and 34.7% relative gains, respectively, when fine-tuning with `Code` compared to `General`. Conversely, the Llama-3-8B model shows a more modest improvement, achieving a 3.6% relative gain. This disparity could be due to differences in the LLMs pretraining, during which models primarily acquire intrinsic knowledge, rather than the IFT stage (Albalak et al. 2024).

Coding data achieves consistent gains as the LLM scales up. To examine the impact of coding data as LLM size increases, we tune Llama-1 and Llama-2 with 7B and 13B parameters under three coding proportion settings. The results are shown in Figure 2.

Coding data tuning consistently enhances LLMs' reasoning capacities across different model families as their size scales up to 13B. This reaffirms the conclusion drawn from smaller model comparisons: coding data can effectively improve LLMs' reasoning capacities during IFT, helping LLMs align better with user needs and handle complex logical intents. The improvement trends from coding data are similar between 13B models and their corresponding 7B models, highlighting the potential of coding data to enhance instruction fine-tuned models on reasoning tasks for larger models.

Tuning with more coding data can better improve the overall reasoning capacities of LLMs. To emphasize the impact of coding data proportions on overall reasoning capacities, we introduce another IFT dataset with 25% coding and 75% natural text, tune Llama-1 and Llama-2 models (7B and 13B parameters) on it, and evaluate each model on all reasoning tasks. Performance trends across 0%, 25%, 50%, and 100% coding data are shown in Figure 3.

`General` Output / `Code` Output	text / text	text / code	code / text	code / code	total
Llama-1	1778	359	83	67	2287
Llama-2	2591	224	63	49	2927

Table 2: **Response format transition statistics.** Count of instances where the `General`-tuned model provides incorrect answers in either text or code format (upper header) and the `Code`-tuned model corrects them (lower header), using Llama-1-13B and Llama-2-13B backbones.

Across different model families and scales, the overall trend of gradual improvement persists after introducing the new 25% coding data setting, reinforcing the findings from Table 4 that coding data plays a key role in evoking the reasoning capabilities of pretrained LLMs. We also observe that performance at 50% coding data is not always better than at 25%, and vice versa, likely due to variance among backbones.

Coding data tuning enhances reasoning in natural text responses. To investigate if coding data tuning helps solve these reasoning tasks by merely relying on producing better code, we examine the presence of code in each response under the condition where the model tuned on `Code` corrects the wrong outcomes of the model tuned on `General`. We employ GPT-3.5-turbo to detect the presence of code in responses and count the number of different answer format transitions (e.g., `General` outputs text and `Code` outputs

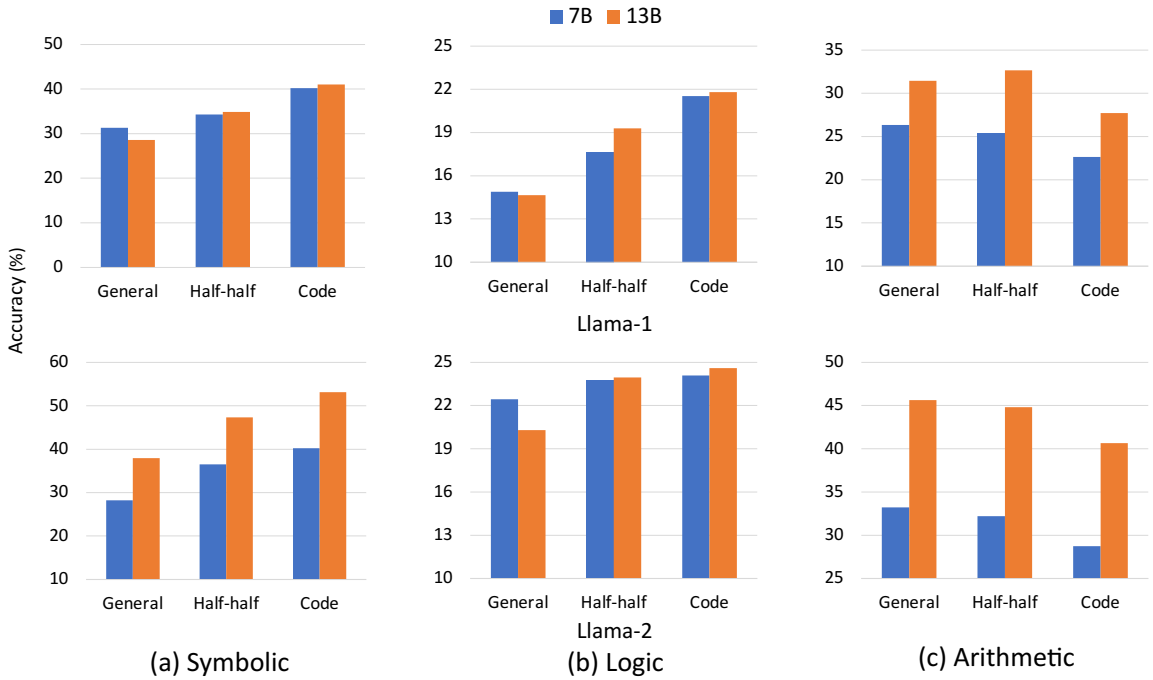

Figure 4: Results comparison under each reasoning domain across different model families.

code). The results with Llama-1-13B and Llama-2-13B as backbone are shown in Table 2. The prompt for determining the response format is in the Appendix.

The `Code`-tuned model *primarily uses pure text responses* to correct answers where the `General`-tuned model is wrong across both model families instead of relying solely on programming skills. Specifically, `Code`-tuned models successfully resolve $\frac{1778+83}{2287} = 81.4\%$ of questions in Llama-1 and $\frac{2591+63}{2927} = 90.7\%$ of questions in Llama-2 using natural text responses. Additionally, `Code`-tuned models do not always use the same answer format as the `General`-tuned model. They automatically choose different formats to obtain correct answers in $\frac{359+83}{2287} = 19.3\%$ and $\frac{224+63}{2927} = 9.8\%$ of cases using Llama-1 and Llama-2, respectively. These results further illustrate that coding data tuning successfully elicits logical thinking in LLMs beyond just in-domain skills, enabling them to answer complex questions in proper formats and improve reasoning task performance.

Results of Different Reasoning Domains

Previous discussions have focused on overall performance across reasoning tasks, but distinct domains require specific skills. For instance, symbolic reasoning tasks like letter concatenation need tokenization skills, which benefit more easily from coding data tuning, while logical reasoning requires multi-hop contextual analysis, and arithmetic tasks need enhanced quantitative reasoning (Yue et al. 2023). To pinpoint the capabilities and limitations of coding data tuning across different domains, we analyze per-domain performance using Llama-1 (Touvron et al. 2023a) and Llama-2 (Touvron et al. 2023b) across various model sizes. Average results across four datasets per domain are shown in Figure 4.

Coding data affects each reasoning ability differently. We observe distinct performance patterns in symbolic, logical, and arithmetic reasoning tasks. For **symbolic reasoning** in Figure 4 (a), the performance improvement from `General` to `Code` tuning is significant and steadily increases. This highlights the effectiveness of code-specific data tuning in enhancing the models' foundational reasoning ability. In **logical reasoning** tasks shown in Figure 4 (b), while all models benefit from more coding data in the IFT dataset, the gains from increasing coding data from 50% to 100% are minor compared to the jump from 0% to 50%. This diminishing return indicates that while coding data enhances reasoning skills, its utility is limited for tasks requiring advanced cognitive functions. For **Arithmetic reasoning** (Figure 4 (c)): The best performance is seen with either `General` or `Half-half`, likely due to the need for models to understand diverse intentions in real-world applications (Cobbe et al. 2021). Tuning with a diverse IFT dataset better meets these needs than coding data alone. Despite this, the performance of `Code`-tuned models remains appealing given the limited data diversity, emphasizing the importance of coding data for LLM tuning.

Within each reasoning domain, models demonstrate similar performance trends. The performance trend within each reasoning domain is consistent across model families, despite variations in architecture and pretraining datasets. This alignment suggests that the underlying factors for eliciting pretrained models' capabilities could be similar within each domain. Our analysis also shows that larger models follow the same trend as their smaller counterparts within the same family, indicating that scalability does not disrupt the effect of coding data during IFT. These findings underscore the potential for the transferability of coding data's effect across

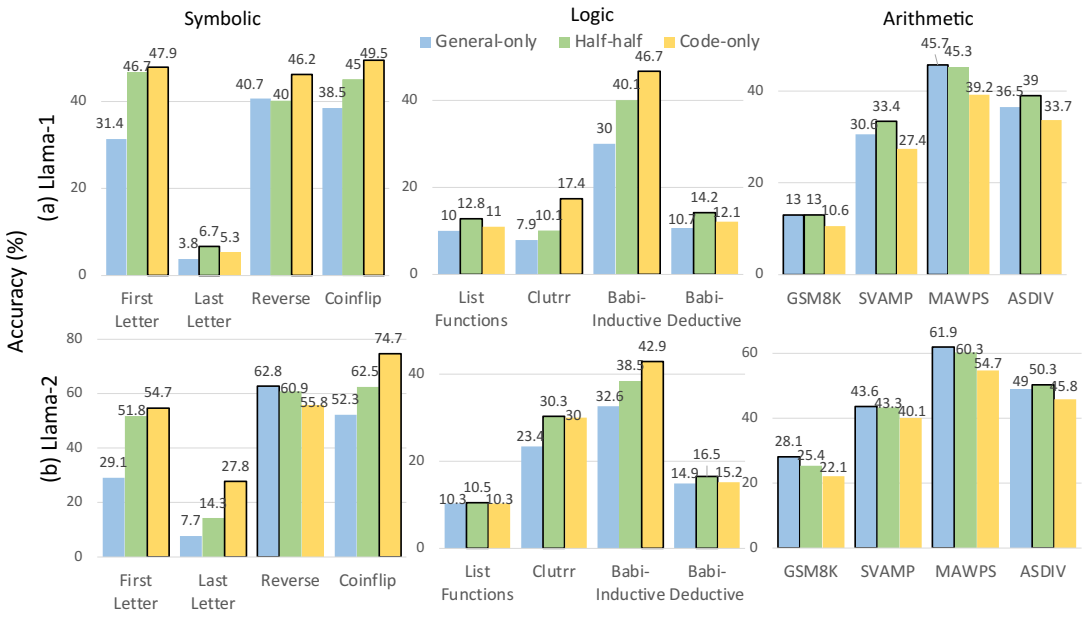

Figure 5: Results comparison for each dataset across on (a) Llama-1-13B and (b) Llama-2-13B.

various LLM backbones and scales at the tuning stage.

	Llama-1			
	text → text	text → code	code → text	code → code
Symbolic	58.9	30.4	4.8	5.9
Logic	93.2	4.8	0.9	1.1
Arithmetic	88.0	6.5	4.9	0.6
	Llama-2			
	text → text	text → code	code → text	code → code
Symbolic	85.5	10.1	1.1	3.3
Logic	96.4	3.0	0.5	0.1
Arithmetic	85.1	8.6	0.8	5.5

Table 3: **Proportions (%) of response transitions across reasoning domains.** Transition types from `General`-tuned to `Code`-tuned models, represented as '`General` (text/code) → `Code` (text/code)', across different reasoning domains for Llama-1-13B and Llama-2-13B.

`Code`-tuned models prefer enhancing reasoning with pure text when the `General`-tuned models output pure text, but preferences diverge across domains when they output coding. We further investigated response transitions where the `Code`-tuned model corrects errors from the `General`-tuned model. Table 3 shows the proportions of transition types from `General` to `Code` for each domain, using Llama-1-13B and Llama-2-13B as backbones.

We find that for both model families, the proportions of text → text' transitions are consistently and significantly higher than those of text → code' across different domains. This reinforces the finding in section 4 that coding data tuning enhances reasoning task performance by genuinely improving reasoning abilities rather than relying solely on in-domain programming skills. Conversely, when `General`-tuned models produce incorrect answers involving coding data, `Code`-tuned models do not show a consistent preference for one format over the other. Specifically, code → text' is preferred in 2 out of 6 settings, while code → code' is preferred in 4. These results demonstrate that `Code`-tuned models can automatically apply appropriate skills to successfully answer questions based on domain properties.

Task-specific Reasoning Capabilities Analysis

The previous subsection highlights domain-level similarities and divergences, which may become more complex at the task level. To investigate this, we delve into each dataset to explore how coding data impacts task-specific reasoning capabilities. Results for models tuned using Llama-1-13B and Llama-2-13B backbones are presented in Figure 5.

Coding data benefits task-specific abilities in Llama-1 and Llama-2 comparatively in symbolic and logic reasoning, but diverges in arithmetic. We observe that models fine-tuned on datasets incorporating coding (`Half-half` and `Code`) demonstrate similar levels of superiority on task-specific abilities across different model families in symbolic and logic reasoning. Specifically, in symbolic tasks, these models either outperform or match their counterparts, `General`, in all four tasks on the Llama-1 and three out of four tasks on the Llama-2, respectively. For logical reasoning, the inclusion of coding data benefits all tasks in both model families. However, the impact of coding data on arithmetic reasoning presents more variation. In the Llama-1, the incorporation of coding data yields better or comparable results across all tasks. Conversely, this benefit is observed only in SVAMP and ASDIV tasks for Llama-2. These results indicate that the benefits of coding data on different LLM backbones are more similar than different, with variations emerging based on the nature of tasks within each subdomain.

The optimal proportion and impact of code vary across tasks. We examine the impact of coding data proportions on

Symbolic Question: Reverse the sequence "chewing gum, magazine".
General-only:
To reverse the order of the words in the given sequence "chewing gum, magazine", we need to follow these steps:
1...
...
6. Return the final string: "gum, magazine".
Code-only:
To reverse the sequence "chewing gum, magazine:
```python
input_str = "chewing gum, magazine"
final_str = ", ".join(reversed(input_str.split(","))) print(final_str)
```
Output:
``` magazine, chewing gum ```

Logic Question: Julius is a lion. Julius is yellow. Brian is a rhino. Brian is gray. Bernhard is a swan. Bernhard is green. Greg is a lion. Lily is a rhino. Greg is white. What color is Lily?
General-only:
It is not possible for me to accurately determine the color of Lily based solely on the provided information...
Code-only:
Lily is a rhino, and the only description we have for a rhino is that it is gray. Therefore, I can conclude that Lily is also gray.

Arithmetic Question: Ryan plants 2 flowers a day in his garden. After 15 days, how many flowers does he have if 5 did not grow?
General-only:
1....
2. ...
Day	Flowers
1	2
2	2
9	5
10	0
14	0
Total = 0 flowers that have grown after 15 days.
3. Therefore, the final answer is 0 flowers that have grown after 15 days.
Code-only:
To answer this question, we need to set up an equation that ...
``` Initially, x = 25 ```
``` x + 25 = 30 ```
``` x + 25 - 5 = 30 -5 = 30 - x ```
``` x = 30 - 5 = 25 ```

Figure 6: **Case study on each reasoning domain (not cherry-picked).** We underline the final answer of each model for clarity and omit some unimportant output due to space limit.

task performance across different model families, finding no single proportion consistently superior for enhancing task-specific reasoning abilities. For Llama-1, the `Half-half` setting achieves or matches the best performance in eight tasks, while four tasks excel with the `Code` setting. For Llama-2, `Half-half` and `Code` perform best in five and four tasks, respectively. Although `Code`-tuned models don't outperform `Half-half`-tuned models in the number of tasks, they often yield greater improvements in specific tasks. For instance, in Llama-1, the `Code` setting shows a significant gain of 16.7 in the Babi-Inductive task, compared to the 3.5 gain with `Half-half` in Babi-Deductive. Similarly, in Llama-2, the `Code` setting achieves an impressive gain of 25.6 in the First Letter task, far exceeding the 7.0 gain with `Half-half` in Babi-Deductive. These findings emphasize the strategic consideration of coding proportions to optimize model effectiveness for specific goals during IFT.

The majority of optimal coding data proportions are consistent across model families. When investigating tasks that benefit from coding data in both model families, we find consistent optimal strategies for most tasks. Specifically, out of 8 tasks analyzed, 5 exhibit the same optimal coding data proportions across model families, while 3 require different strategies. This suggests that the proportion of coding data in IFT datasets can similarly enhance task-specific reasoning abilities across different model families, indicating a foundational influence of coding data that is generally model-agnostic. However, variability in the remaining tasks underscores the need for flexible adaptation strategies for each model family.

Case Study

To show how coding data tuning corrects responses for different reasoning tasks, we focus on instances where the `Code`-tuned model succeeds while the `General`-tuned model fails, on Llama-1-13B backbone. We randomly select one instance from each reasoning domain in Figure 6.

For the symbolic task, the `Code`-tuned model uses Python code to correctly reverse the sequence, while the `General`-tuned model, confused by phrase construction, fails to execute the reverse action, resulting in an incorrect answer. In the logic task, the `General`-tuned model fails to follow the inductive reasoning path required for the correct answer. Conversely, the `Code`-tuned model delivers a concise and accurate answer, demonstrating improved logical capacity due to coding data tuning. For the arithmetic question, both models use structural outputs to answer, but the `General`-tuned model generates an incorrect structural reasoning path, resulting in a wrong answer. Conversely, the `Code`-tuned model leverages its in-domain coding capacity by outputting structural math equations to obtain the correct final answer. These case studies showcase how coding data tuning enhances LLM reasoning capacities to properly answer diverse questions.

5 Conclusion

We studied the impact of coding data on LLMs' reasoning capacities during the IFT stage using a multi-perspective approach. Resource limitations are prevented by examining larger models like Llama-2/3 70B and we conduct evaluations focused on generative benchmark tasks within three reasoning domains. We highlighted the general effects of coding data tuning, suggesting that future research should explore detailed aspects such as diverse coding data types, content formats, and low-quality coding data. We hope this work inspires further research on LLMs' reasoning, including instruction fine-tuning, evaluation, and analysis.
```

Acknowledgments

We gratefully acknowledge financial support by the National Institutes for Health (NIH) grant NIH 7R01HL149670.

References

AI@Meta. 2024. Llama 3 Model Card.

Albalak, A.; Elazar, Y.; Xie, S. M.; Longpre, S.; Lambert, N.; Wang, X.; Muennighoff, N.; Hou, B.; Pan, L.; Jeong, H.; Raffel, C.; Chang, S.; Hashimoto, T.; and Wang, W. Y. 2024. A Survey on Data Selection for Language Models. arXiv:2402.16827.

Anthropic. 2023. Claude 2.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; Hui, B.; Ji, L.; Li, M.; Lin, J.; Lin, R.; Liu, D.; Liu, G.; Lu, C.; Lu, K.; Ma, J.; Men, R.; Ren, X.; Ren, X.; Tan, C.; Tan, S.; Tu, J.; Wang, P.; Wang, S.; Wang, W.; Wu, S.; Xu, B.; Xu, J.; Yang, A.; Yang, H.; Yang, J.; Yang, S.; Yao, Y.; Yu, B.; Yuan, H.; Yuan, Z.; Zhang, J.; Zhang, X.; Zhang, Y.; Zhang, Z.; Zhou, C.; Zhou, J.; Zhou, X.; and Zhu, T. 2023. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Chen, L.; Li, S.; Yan, J.; Wang, H.; Gunaratna, K.; Yadav, V.; Tang, Z.; Srinivasan, V.; Zhou, T.; Huang, H.; and Jin, H. 2023. AlpaGasus: Training A Better Alpaca with Fewer Data. arXiv:2307.08701.

Chen, Z. Z.; Ma, J.; Zhang, X.; Hao, N.; Yan, A.; Nourbakhsh, A.; Yang, X.; McAuley, J.; Petzold, L.; and Wang, W. Y. 2024. A Survey on Large Language Models for Critical Societal Domains: Finance, Healthcare, and Law. arXiv:2405.01769.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Chung, H. W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, Y.; Wang, X.; Dehghani, M.; Brahma, S.; Webson, A.; Gu, S. S.; Dai, Z.; Suzgun, M.; Chen, X.; Chowdhery, A.; Castro-Ros, A.; Pellat, M.; Robinson, K.; Valter, D.; Narang, S.; Mishra, G.; Yu, A.; Zhao, V.; Huang, Y.; Dai, A.; Yu, H.; Petrov, S.; Chi, E. H.; Dean, J.; Devlin, J.; Roberts, A.; Zhou, D.; Le, Q. V.; and Wei, J. 2022. Scaling Instruction-Finetuned Language Models. arXiv:2210.11416.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Fortes, A. 2023. Simple Dataset Generation. https://github.com/atfortes/simple-dataset-generation.

Fu, H., Yao; Peng; and Khot, T. 2022. How does GPT Obtain its Ability? Tracing Emergent Abilities of Language Models to their Sources. *Yao Fu's Notion*.

Guo, D.; Zhu, Q.; Yang, D.; Xie, Z.; Dong, K.; Zhang, W.; Chen, G.; Bi, X.; Wu, Y.; Li, Y. K.; Luo, F.; Xiong, Y.; and Liang, W. 2024. DeepSeek-Coder: When the Large Language Model Meets Programming – The Rise of Code Intelligence. arXiv:2401.14196.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; and Hajishirzi, H. 2016. MAWPS: A Math Word Problem Repository. In Knight, K.; Nenkova, A.; and Rambow, O., eds., *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1152–1157. San Diego, California: Association for Computational Linguistics.

Li, S.; Chen, J.; Shen, Y.; Chen, Z.; Zhang, X.; Li, Z.; Wang, H.; Qian, J.; Peng, B.; Mao, Y.; Chen, W.; and Yan, X. 2022. Explanations from Large Language Models Make Small Reasoners Better. arXiv:2210.06726.

Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A.; Newman, B.; Yuan, B.; Yan, B.; Zhang, C.; Cosgrove, C.; Manning, C. D.; Ré, C.; Acosta-Navas, D.; Hudson, D. A.; Zelikman, E.; Durmus, E.; Ladhak, F.; Rong, F.; Ren, H.; Yao, H.; Wang, J.; Santhanam, K.; Orr, L.; Zheng, L.; Yuksekgonul, M.; Suzgun, M.; Kim, N.; Guha, N.; Chatterji, N.; Khattab, O.; Henderson, P.; Huang, Q.; Chi, R.; Xie, S. M.; Santurkar, S.; Ganguli, S.; Hashimoto, T.; Icard, T.; Zhang, T.; Chaudhary, V.; Wang, W.; Li, X.; Mai, Y.; Zhang, Y.; and Koreeda, Y. 2023. Holistic Evaluation of Language Models. arXiv:2211.09110.

Longpre, S.; Hou, L.; Vu, T.; Webson, A.; Chung, H. W.; Tay, Y.; Zhou, D.; Le, Q. V.; Zoph, B.; Wei, J.; and Roberts, A. 2023. The Flan Collection: Designing Data and Methods for Effective Instruction Tuning. arXiv:2301.13688.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023a. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. arXiv:2308.09583.

Luo, Z.; Xu, C.; Zhao, P.; Sun, Q.; Geng, X.; Hu, W.; Tao, C.; Ma, J.; Lin, Q.; and Jiang, D. 2023b. WizardCoder: Empowering Code Large Language Models with Evol-Instruct. arXiv:2306.08568.

Ma, Y.; Liu, Y.; Yu, Y.; Zhang, Y.; Jiang, Y.; Wang, C.; and Li, S. 2023. At Which Training Stage Does Code Data Help LLMs Reasoning? *arXiv preprint arXiv:2309.16298*.

Madaan, A.; Zhou, S.; Alon, U.; Yang, Y.; and Neubig, G. 2022. Language Models of Code are Few-Shot Commonsense Learners. arXiv:2210.07128.

Miao, S.-y.; Liang, C.-C.; and Su, K.-Y. 2020. A Diverse Corpus for Evaluating and Developing English Math Word Problem Solvers. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting*

of the Association for Computational Linguistics, 975–984. Online: Association for Computational Linguistics.

OpenAI. 2022. Introducing ChatGPT. Accessed: 2023-05-11.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; Schulman, J.; Hilton, J.; Kelton, F.; Miller, L.; Simens, M.; Askell, A.; Welinder, P.; Christiano, P.; Leike, J.; and Lowe, R. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Peng, B.; Li, C.; He, P.; Galley, M.; and Gao, J. 2023. Instruction Tuning with GPT-4. *arXiv preprint arXiv:2304.03277*.

Qiu, L.; Jiang, L.; Lu, X.; Sclar, M.; Pyatkin, V.; Bhagavatula, C.; Wang, B.; Kim, Y.; Choi, Y.; Dziri, N.; et al. 2023. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. *arXiv preprint arXiv:2310.08559*.

Rule, J. S. 2020. *The child as hacker: building more human-like models of learning*. Ph.D. thesis, Massachusetts Institute of Technology.

Sharegpt. 2023. ShareGPT.

Sinha, K.; Sodhani, S.; Dong, J.; Pineau, J.; and Hamilton, W. L. 2019. CLUTRR: A Diagnostic Benchmark for Inductive Reasoning from Text. *CoRR*, abs/1908.06177.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Team, G.; Mesnard, T.; Hardin, C.; Dadashi, R.; Bhupatiraju, S.; Pathak, S.; Sifre, L.; Rivière, M.; Kale, M. S.; Love, J.; et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023a. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I.; Korenev, A.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X. E.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023b. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Wang, X.; Li, S.; and Ji, H. 2023. Code4Struct: Code Generation for Few-Shot Event Structure Prediction. arXiv:2210.12810.

Wason, P. C. 1968. Reasoning about a rule. *Quarterly journal of experimental psychology*, 20(3): 273–281.

Wason, P. C.; and Johnson-Laird, P. N. 1972. *Psychology of reasoning: Structure and content*, volume 86. Harvard University Press.

Wei, J.; Bosma, M.; Zhao, V. Y.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2022a. Finetuned Language Models Are Zero-Shot Learners. arXiv:2109.01652.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; Chi, E. H.; Hashimoto, T.; Vinyals, O.; Liang, P.; Dean, J.; and Fedus, W. 2022b. Emergent Abilities of Large Language Models. arXiv:2206.07682.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; hsin Chi, E. H.; Le, Q.; and Zhou, D. 2022c. Chain of Thought Prompting Elicits Reasoning in Large Language Models. *ArXiv*, abs/2201.11903.

Weston, J.; Bordes, A.; Chopra, S.; Rush, A. M.; Van Merriënboer, B.; Joulin, A.; and Mikolov, T. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Xu, C.; Sun, Q.; Zheng, K.; Geng, X.; Zhao, P.; Feng, J.; Tao, C.; and Jiang, D. 2023. WizardLM: Empowering Large Language Models to Follow Complex Instructions. arXiv:2304.12244.

Yang, X.; Li, Y.; Zhang, X.; Chen, H.; and Cheng, W. 2023. Exploring the Limits of ChatGPT for Query or Aspect-based Text Summarization. arXiv:2302.08081.

Yuan, Z.; Liu, J.; Zi, Q.; Liu, M.; Peng, X.; and Lou, Y. 2023. Evaluating instruction-tuned large language models on code comprehension and generation. *arXiv preprint arXiv:2308.01240*.

Yue, X.; Qu, X.; Zhang, G.; Fu, Y.; Huang, W.; Sun, H.; Su, Y.; and Chen, W. 2023. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. *arXiv preprint arXiv:2309.05653*.

Zhang, X.; Li, S.; Yang, X.; Tian, C.; Qin, Y.; and Petzold, L. R. 2023. Enhancing Small Medical Learners with Privacy-preserving Contextual Prompting. arXiv:2305.12723.

Zhang, X.; Tian, C.; Yang, X.; Chen, L.; Li, Z.; and Petzold, L. R. 2024. AlpaCare:Instruction-tuned Large Language Models for Medical Application. arXiv:2310.14558.