

ICLEval: Evaluating In-Context Learning Ability of Large Language Models

Wentong Chen¹, Yankai Lin^{1*}, ZhenHao Zhou², HongYun Huang²,
Yantao Jia², Zhao Cao², Ji-Rong Wen^{1,3,4}

¹Gaoling School of Artificial Intelligence, Renmin University of China

²Huawei Poisson Lab

³Beijing Key Laboratory of Big Data Management and Analysis Methods

⁴School of Information, Renmin University of China

Abstract

In-Context Learning (ICL) is a critical capability of Large Language Models (LLMs) as it empowers them to comprehend and reason across interconnected inputs. Evaluating the ICL ability of LLMs can enhance their utilization and deepen our understanding of how this ability is acquired at the training stage. However, existing evaluation frameworks primarily focus on language abilities and knowledge, often overlooking the assessment of ICL ability. In this work, we introduce the ICLEval benchmark to evaluate the ICL abilities of LLMs, which encompasses two key sub-abilities: exact copying and rule learning. Through the ICLEval benchmark, we demonstrate that ICL ability is universally present in different LLMs, and model size is not the sole determinant of ICL efficacy. Surprisingly, we observe that ICL abilities, particularly copying, develop early in the pretraining process and stabilize afterward. Our source codes and benchmark are released at <https://github.com/RUCBM/ICLEval>.

1 Introduction

Large Language Models (LLMs) possess stronger language abilities and richer knowledge compared to traditional models, and one of their most important differentiating factors is their In-Context Learning (ICL) ability (Brown et al., 2020). ICL ability serves as a hallmark of LLMs and plays a crucial role in their strong generalization performance. It enables LLMs to quickly adapt to new tasks without altering their internal parameters, utilizing techniques like zero-shot and few-shot learning. Moreover, ICL is a pivotal component in tool learning (Qin et al., 2023b) and various reasoning strategies, such as chain-of-thought (Wei et al., 2022) and tree-of-thought (Yao et al., 2023).

The advantages of ICL have attracted many researchers to explore how and why it works (Lu et al., 2021; Nie et al., 2022; Ye et al., 2022; Min

et al., 2022; Liu et al., 2023a). Some researchers (Olsson et al., 2022) find that induction heads can implement pattern copying behavior and appear to be the primary source of ICL. Pattern copying behavior means LLMs can exactly copy the previous content based on prefix matching or generate similar content based on learned rules. A natural idea is to do a quantitative evaluation including both aspects, to help better explore the sources and influencing factors of LLMs' ICL ability. Currently, evaluations of LLMs primarily focus on language abilities (Reddy et al., 2019; Lai et al., 2017; Rajpurkar et al., 2018), knowledge applications (Clark et al., 2018; Bisk et al., 2020; Zellers et al., 2019; Mihaylov et al., 2018), and complex abilities (Yuan et al., 2023; Chen et al., 2021a; Hendrycks et al., 2020; Huang et al., 2023). Utilizing these data as ICL tasks may introduce interference and hinder the assessment of LLMs' real ICL abilities.

In this work, we introduce the ICLEval benchmark to systematically evaluate the ICL ability of LLMs. Given that pattern copying behavior can be regarded as the main performance of ICL, we construct our tasks from two aspects: exact copying and rule learning. Exact copying means LLMs match the same prefix and copy the subsequent content, while rule learning means LLMs learn the rule from examples and complete similar content. We incorporate two scenarios (unstructured and structured contexts) to evaluate exact copying ability and consider various scenarios of rules (format, order, statistics, etc.) to assess rule learning ability.

We conduct experiments to explore the factors that affect ICL ability. Firstly, we consider if the **model size** is the key factor that affects the ICL ability. We test a series of LLMs ranging from 1.1B to 65B parameters. We observe that larger models often exhibit stronger ICL ability, but some smaller models can also be compared with the larger ones. Secondly, we explore how the ICL ability changes in the **pretraining stage** by testing the checkpoints

*Corresponding Author

Type	Task Name	Num.
Exact Copying		
Unstructured Text	String Completion	100
Structured Text	Dictionary Search	190
Rule Learning		
	Format Check	120
Format Rules	Format Cloning	100
	Format Conversion	120
	Order Check	100
Order Rules	Order Adjustment	240
	Duplication Check	300
	De-Duplication	300
Statistics Rules	Count & Navigation	120
	Relation Analysis	100
	List Mapping	Numbers' List Mapping
Total		2,040

Table 1: The number of samples in our ICL tasks.

with different training tokens. Surprisingly, we find that most ICL abilities reach their peak in the early stages of pretraining, with minimal growth during subsequent training. Finally, we analyze some bad cases from four aspects.

2 Benchmark Construction

In-context learning is an inherent capability of LLMs, enabling them to perform various tasks without modifying their parameters. However, evaluating ICL directly presents challenges as it can be easily influenced by language proficiency and knowledge. For instance, comparing the ICL ability of two different models on common tasks like the GLUE benchmark (Wang et al., 2018) is challenging, even when using the same n-shot settings. If one model obtains a lower score, it could be attributed to a weakness in ICL ability or language understanding, making it difficult to determine the exact cause. In this study, we aim to create evaluation scenarios for ICL that mitigate potential confounding factors.

ICL primarily hinges on two foundational skills: (1) the exact copying ability, which can be represented as " $AB \dots A \rightarrow B$ " models need to perform two steps: first discriminate the "A" part, as referred to prefix matching and then copy the "B" part. (2) the rule learning ability, which enables the model to identify patterns and rules within the aggregated information derived from the n-shot examples, which are relevant to the task at hand. Therefore, we design several tasks for evaluating the copy and learning abilities in ICLEval. The categories and statistics for each task are presented in Table 1, and more details are in Appendix A.

2.1 Exact Copying Ability

Copying is a fundamental ability of the ICL mechanism, allowing LLMs to gather supplementary information from contextual cues. In our evaluation, we design tasks specifically to assess the exact copying ability of LLMs, wherein they are required to copy fragments that are present within the given context. To evaluate the models' adaptability and proficiency across various contextual scenarios, ICLEval categorizes copying tasks into two distinct types: unstructured (natural language) text and structured text.

Copying in unstructured text. As shown in the left-top example of Figure 1, we mask the second half of a string (str. "521351caba13f591") which appears one or multiple times in the previous paragraphs. Then we require the models to predict it given the first half (str. "521351ca"). Notably, we use the hash strings to replace the origin entities in order to prevent LLMs from completing the entity based on their internal knowledge rather than the context.

Copying in structured text. We use the "dictionary" format as a representation of structured data to better control the length and similarity of "A" part and "B" part. As shown in the right-top example of Figure 1, this task challenges the models to deduce a value from a specified key (str. "71df7d") from a set of key-value pairs. We test their ability to seek and extract information from structured repositories efficiently.

2.2 Rule Learning Ability

The learning ability is another fundamental ability of the ICL mechanism, allowing LLMs to extract mapping rules from in-context examples. This ability enables them to tackle different and unseen tasks based on examples or natural language descriptions, without requiring updates to model parameters. To focus solely on the learning ability and avoid the influence of language ability and knowledge across different models, ICLEval decomposes the evaluation of learning ability into four foundational aspects: format rules, order rules, statistics rules, and list mapping.

Learning format rules assesses whether LLMs can learn formatting mappings from in-context examples, a key skill for generating appropriately formatted outputs across various tasks. Specifically, we design a format classification task and two for-

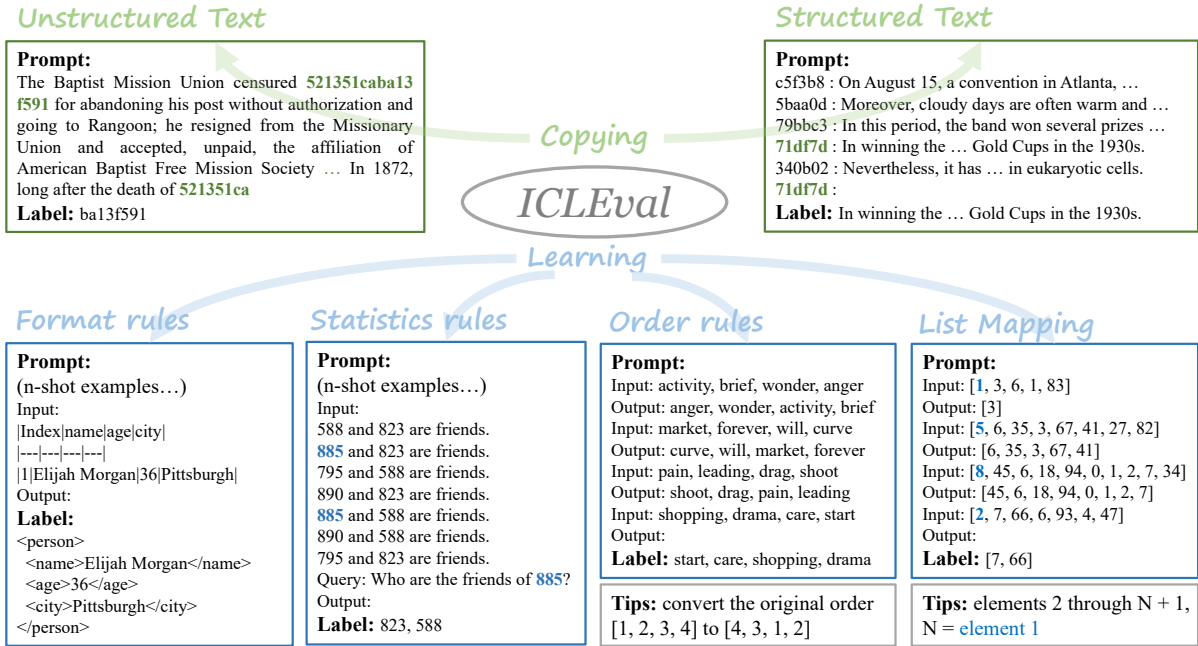


Figure 1: The examples of six representative tasks in ICLEval.

mat generation tasks named format check, format cloning, and format conversation respectively. Figure 1 shows an example of the format conversation task that converts "Markdown-Table" format data to "XML" format.

Learning order rules examines LLMs' ability to grasp the order of a group of elements as well as the mapping rule of two groups of elements from in-context examples. This skill is crucial for tasks requiring re-organization of input elements, such as in translation and syntactic analysis scenarios. To this end, we formulate a classification task and a generation task aimed at evaluating LLMs' capabilities in determining whether inputs satisfy a specific order (named order check) and in executing order transformations (named order adjustment). Figure 1 shows an example of the order adjustment task that converts the word-level order from [1, 2, 3, 4] to [4, 3, 1, 2].

Learning statistics rules evaluates LLMs' ability to extract, filter, summarize, and analyze relevant information from in-context examples. We consider four typical tasks including detecting or eliminating duplicates within the context (named duplication check and de-duplication), counting the number of elements in the context (named count & navigation), and generating the related information about a given node in relation graph (named relation analysis). These tasks collectively aim to measure LLMs' analytical capabilities and their ap-

plications in processing and interpreting complex information structures. Figure 1 shows an example of the relation analysis task that lists all the friends of "885" mentioned in the context.

Learning list mapping is to find diverse custom rules from given in-context examples. We use the data from "list_functions" task in BIGBench (Srivastava et al., 2022) to compose our numbers' list mapping task, which is learning a mapping given multi-groups of numbers list pairs. There are 250 diversity mapping rules in it with different difficulties, and one example is shown in Figure 1.

All these rule learning tasks we designed utilize n-shot examples while relying less on language abilities, commonsense knowledge, or factual knowledge.

3 Experiments

In the experiments, we first list the models evaluated in our work. Then, we analyze the effects of model size and pretraining tokens. Later, we give the analysis of four other factors impacting in-context learning ability. More detailed results are provided in Appendix C.2.

3.1 Settings

Metrics We have a total of 12 tasks with 2,040 testing samples. For almost all tasks such as string completion, dictionary search, and format conversion, we use exact match scores to evaluate the pre-

Model	Exact Copying		Rule Learning				Average
	Unstructured Text	Structured Text	Format	Order	Statistics	List Mapping	
TinyLlama-1.1B	<u>0.89</u>	0.46	<u>0.60</u>	0.46	0.23	0.36	<u>0.50</u>
Phi-1.5	0.54	0.25	0.48	0.46	<u>0.25</u>	<u>0.39</u>	0.40
GPT-Neo-1.3B	0.54	<u>0.48</u>	0.48	<u>0.50</u>	0.20	0.38	0.43
ChatGLM3-6B	0.90	0.70	0.60	0.54	0.32	0.43	0.58
Qwen-7B	0.89	0.45	0.58	0.62	0.42	0.48	0.57
Baichuan2-7B	0.87	0.44	0.63	0.53	0.34	0.49	0.55
LLaMA-7B	0.87	0.33	0.68	0.54	0.27	0.42	0.52
LLaMA2-7B	0.89	0.45	0.61	0.63	0.32	0.45	0.56
LLaMA3-8B	0.57	<u>0.87</u>	<u>0.69</u>	0.94	<u>0.61</u>	<u>0.63</u>	<u>0.72</u>
Mistral-7B-v0.1	0.88	0.54	0.67	0.78	0.46	0.56	0.65
Qwen-14B	0.87	0.48	0.68	0.79	0.54	0.55	0.65
Baichuan2-13B	0.80	0.23	<u>0.75</u>	0.66	0.40	0.39	0.54
LLaMA-13B	<u>0.89</u>	0.43	0.63	0.55	0.32	0.47	0.55
LLaMA2-13B	<u>0.89</u>	0.47	0.60	0.72	0.36	0.52	0.59
Mixtral-8x7B	0.88	<u>0.88</u>	0.71	<u>0.86</u>	<u>0.63</u>	<u>0.59</u>	<u>0.76</u>
LLaMA-30B	<u>0.89</u>	0.33	0.73	0.77	0.46	0.53	0.62
LLaMA-65B	<u>0.89</u>	0.49	0.59	0.85	0.47	0.58	0.65
LLaMA2-70B	<u>0.89</u>	0.43	0.75	0.87	0.59	0.58	0.69
LLaMA3-70B	0.64	0.91	0.90	0.94	0.87	0.66	0.82

Table 2: Performance of LLMs with different sizes and the last column is the mean score of all aspects. A score with underline indicates the highest score in the same group of models, while a score with **bold** indicates the highest score among all the evaluated models.

dictionaries with the labels. But for the format cloning task, we only evaluate the correctness of the format and do not consider the content. Moreover, we use postprocessing to convert models’ responses for different tasks. More processing details are shown in Appendix A.3.

Evaluated Models We evaluate various open-sourced LLMs with different model sizes, such as the LLaMA series (Touvron et al., 2023a,b), the Baichuan series (Baichuan, 2023), and the Qwen series (Bai et al., 2023). For example, we evaluate the LLaMA series with 7B, 13B, 34B, and 65B versions as well as base-version and chat-version. Moreover, we test the intermediate checkpoints with different pretraining stages for TinyLlama-1.1B, Baichuan-7B, and Amber-7B. The detailed description of these models can be found in Appendix B.

Inference Settings For all the models, whether base-version or chat-version, we don’t use additional prompts (e.g. "User:"), and use n-shot examples in most tasks. We don’t use sampling or beam search and only use the greedy decoding method.

3.2 How do Model Sizes Effect ICL Ability?

We divided the models into three groups: small-sized models (around 1B parameters), middle-sized models (around 10B parameters), and large-sized

models (bigger than 30B parameters), and conducted tests on all these models. The results are presented in Table 2. From the table, we can obtain the following research findings:

(1) A general trend indicates that larger models tend to exhibit superior ICL performance, observing the LLaMA series from 7B to 65B. However, the data also presents a considerable variance within models of similar sizes (e.g. LLaMA-7B, LLaMA2-7B, and LLaMA3-8B). Notably, some middle-sized models (e.g. such as Mistral-7B-v0.1 and LLaMA3-8B) demonstrate strong ICL abilities comparable to models having 5x-10x size (e.g. LLaMA-65B). This finding underscores that model size is not the sole determinant of ICL efficacy.

(2) For the exact copying ability, larger models don’t have obvious advantages. Surprisingly, in the unstructured context scenario, even the small-sized models (e.g. TinyLlama-1.1B) can achieve a score of 0.89, while the scores of LLaMA3-8B and LLaMA3-70B are only 0.57 and 0.64 respectively. We find that the results are mainly influenced by tokenizers, which we have a detailed discussion in Section 3.4 and Figure 10. In the structured context scenario, the results are also unrelated to the models’ sizes. With similar model sizes, LLaMA-7B gets 0.33 but LLaMA3-8B gets 0.87, LLaMA2-70B gets 0.43 but LLaMA3-70B gets 0.91.

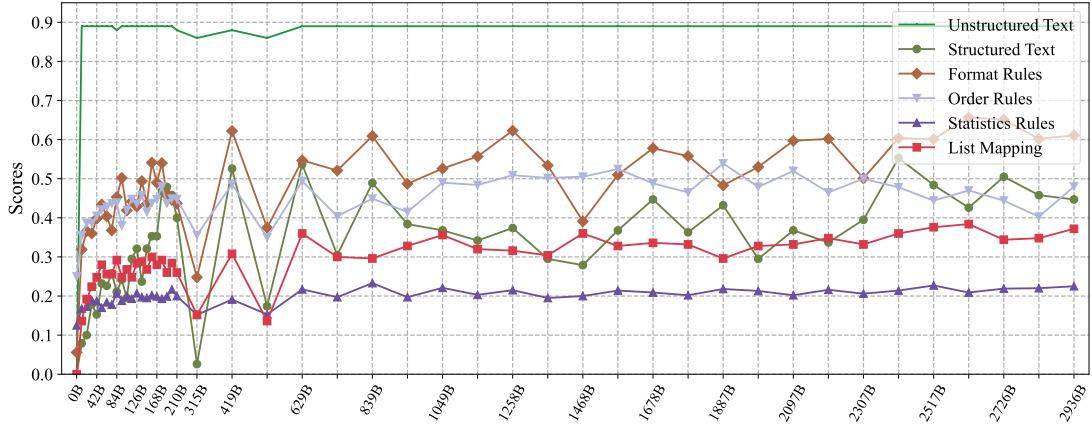


Figure 2: The scores in the pretraining stage of TinyLlama-1.1B with 3T tokens.

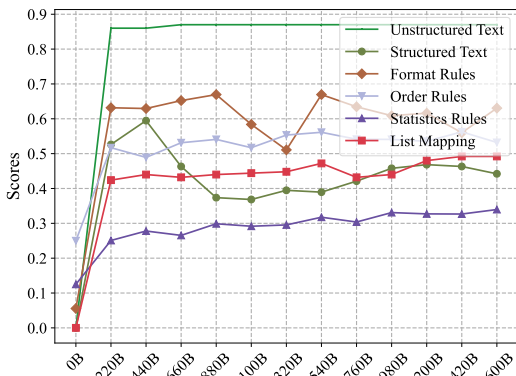


Figure 3: The scores in the pretraining stage of Baichuan2-7B with 2.6T tokens.

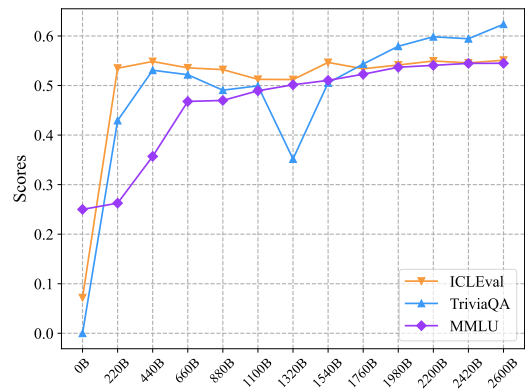


Figure 4: Baichuan2-7B’s ICL ability and knowledge. The ICL ability of the model is acquired in the early stage of pretraining, while the knowledge is acquired in the whole pretraining stage.

(3) For the rule learning ability, larger models usually have better performance than smaller ones. For challenging rules such as statistics and list mapping, the gaps between small-sized, middle-sized, and large-sized models become more obvious. The highest scores of the three groups of models are 0.25 v.s. 0.63, v.s. 0.87 in statistics rules and 0.39 v.s. 0.63, v.s. 0.66 in list mapping rules. Especially, the largest model in our evaluation (LLaMA3-70B) gets all the highest scores of the rule learning tasks.

3.3 How does the ICL Ability Changing in the Pretraining Stage?

We examine how the ICL ability evolves during the pretraining stage with the pretraining checkpoints of TinyLlama-1.1B, Baichuan2-7B, and Amber-7B. Figure 2 and Figure 3 show the detailed results of TinyLlama-1.1B and Baichuan2-7B, and the results of Amber-7B are shown in Appendix C.3. In addition, we calculate the average scores of Baichuan2-7B and test the TriviaQA (Joshi et al., 2017) and

MMLU (Hendrycks et al., 2020) datasets with the same checkpoints, as shown in Figure 4.

(1) The abilities of ICL exhibit rapid growth in the initial stage, before about 200B tokens. However, after this point, their growth becomes slow and eventually stops. Figure 4 illustrates the results of ICLEval, TriviaQA, and MMLU, which represent the model’s abilities in terms of ICL, knowledge, and processing complex questions, respectively. We can see that after 220B tokens, the ICL abilities nearly no increase, while knowledge of models continued to increase. The ability to process complex questions demonstrates little improvement before training 220B tokens but experiences a significant boost afterward. This phenomenon indicates that LLMs acquire different abilities in a sequential order, with ICL ability being relatively easy to obtain.

(2) The exact copying ability emerges in the very early stages of pretraining. As shown in Figure 2,

the result of copying in **unstructured text** scenarios arrives at the highest before 10B tokens and remains stable until the end of pretraining. Besides, the result of copying in **structured text** scenarios arrives at the highest score of about 0.53 at the 419B tokens. It is unstable and fluctuates multiple times during the subsequent training stages. Compared with the unstructured text scenarios, we design various similar keys as the interference factors of prefix matching in structured text scenarios. We suppose that the distinguishing ability of similar strings is unstable in the pertaining stage.

(3) The rule learning ability increases very slowly after the initial stage, about 200B training tokens. Combined with Figure 2 and Figure 3, we can find that the results of learning format rules are unstable in the pertaining stage, while the results of the other three aspects rules are more stable. We speculate that format learning is more easily interference by the LLMs’ inherent preferences, and these inherent preferences continuously change during the training stage. Inherent preferences will be discussed in Section 3.4.

In particular, there is a significant increase in statistics for Baichuan-7B but only a slight improvement in TinyLlama-1.1B. We guess that maybe models only learn to predict the next token based on a small number of previous tokens in the early stages of pretraining. While, during the later stage of pretraining, models learn to use more tokens to predict the next token more accurately. Small-sized models may not have enough attention points capacity for later pretraining, which will be discussed in detail in Section 3.4.

3.4 Case Study.

We find some interesting phenomenons in our evaluation results and regard these phenomena to four aspects: distinguishing ability, inherent preferences, attention points capacity, and tokenizer. We also analyze these phenomena with some bad cases. Due to the limitation of pages, we put the bad cases on the Appendix C.1.

Distinguishing Ability Similar strings can cause interference for humans as well as for LLMs. As depicted in Figure 5, in the dictionary search task, the accuracy of the same model is adversely affected when there are numerous similar keys in the dictionary. More similar keys make the scores drop more, while models with the stronger distinguishing ability (e.g. Baichuan2-7B) drop less. The

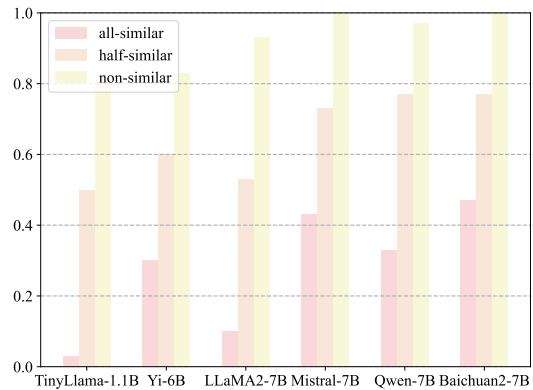


Figure 5: Performance changes when there are more similar strings in the in-context examples.

results suggest that the excessive presence of similar strings makes the model chaotic, causing it to struggle to distinguish between different keys. This phenomenon indicates that models with weak distinguishing ability may make mistakes when extracting information in complex in-context environments.

We looked into the reasons for errors in certain cases and found two types. First, the model sometimes selects the incorrect value from the dictionary. Second, it occasionally generates a random string that is not present in the dictionary. We further analyze the proportion of right results and the two error types in the pretraining stage of Baichuan2-7B, as shown in Figure 6. We find the model’s choices of the two types are continuously changing. That indicates the distinguishing ability is unstable in the pretraining stage.

Inherent Preferences We observe that some models exhibit unusually weak performance in tasks such as format check or format cloning. Figure 7 shows the performance of the format check task. This task is a classification task with six different labels: "JSONL", "CSV", "Tuple", "YAML", "Table", and "XML". From the figure, we can find that ChatGLM3-6B-Chat can get a score approaching 0.7, while lots of models are lower than the random scores. To our surprise, LLaMA-65B gets 0.0 in this task.

We further give a deeper analysis of the bad cases, and we find that most models tend to respond with "JSON" as their prediction, but we even don’t have this label. We suppose that such heavy inherent preferences of these models may come from their pretraining data distribution, making the model cannot adhere to the pre-defined formats or

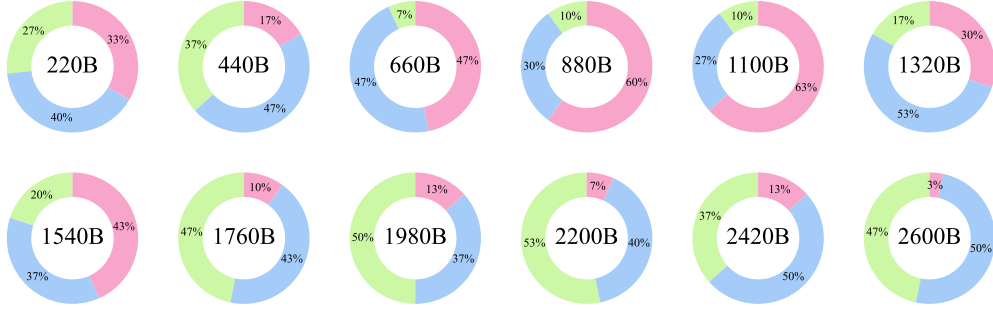


Figure 6: The proportion of three answer types in the pretraining stage of Baichuan2-7B. Green means copying right value, blue means copying incorrect values, and pink means generating random values. We find the model’s choices are continuously changing in the pretraining stage.

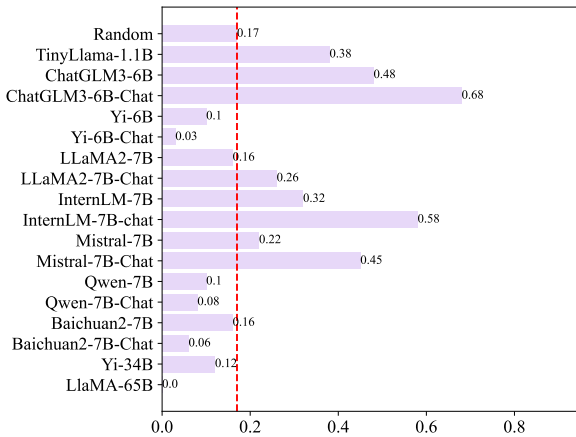


Figure 7: Performance of format check task.

rules presented in the in-context examples.

Furthermore, we find the chat version of some models such as ChatGLM3-6B, InternLM-7B, and Mistral-7B have obvious improvements to their base version in this task. We think this might be due to the instruction learning process can reduce the impact of the models’ inherent preferences to some extent.

Attention Points Capacity In the count & navigation task, the "1-dim" setting requires models to analyze and count only two elements ("up" and "down"), whereas the "2-dim" setting involves analyzing and counting four elements ("up", "down", "right" and "left"). We have observed that the scores for the "1-dim" setting are significantly higher than those for the "2-dim" setting, as illustrated in Table 3.

We believe this phenomenon occurs because models face difficulties in effectively utilizing a larger number of tokens within the context to predict the next token. In other words, for models

Model	Count		Navigation	
	1-dim	2-dim	1-dim	2-dim
TinyLlama-1.1B	0.17	0.03	0.27	0.03
Yi-6B	0.40	0.00	0.43	0.03
LLaMA2-7B	0.47	0.07	0.27	0.13
Mistral-7B-v0.1	0.77	0.43	0.47	0.10
Qwen-7B	0.60	0.10	0.53	0.10
Baichuan2-7B	0.77	0.07	0.33	0.07

Table 3: The scores of count & navigation task. We split the tasks by task types.

to accurately predict results, they need to pay attention to multiple points within the context. And we call this Attention Points Capacity. However, many models tend to rely on only a few tokens for predicting the next token, thereby failing to incorporate all the available information simultaneously. To validate our hypothesis, we conducted a further analysis of the relationship between the number of elements and accuracy, as depicted in Figure 8. The figure demonstrates a significant decrease in accuracy as the number of elements increases.

Tokenizer In our evaluation, we observe that the tokenizer has a significant impact on our results and can affect our adjustment of task difficulty. From a human perspective, we can perceive text at the character level, word level, or sentence level, allowing us to easily distinguish individual numbers, letters, words, and sentences. However, language models process inputs at the token level, which presents a different viewpoint compared to ours.

In Figure 9, we illustrate three types of bad cases caused by tokenization that we find can lead to confusion for LLMs. "Token fracture" occurs when a continuous sequence is split but the encoded tokens of the segmented fragments do not match those of the original sequence; "Token insert" refers to the

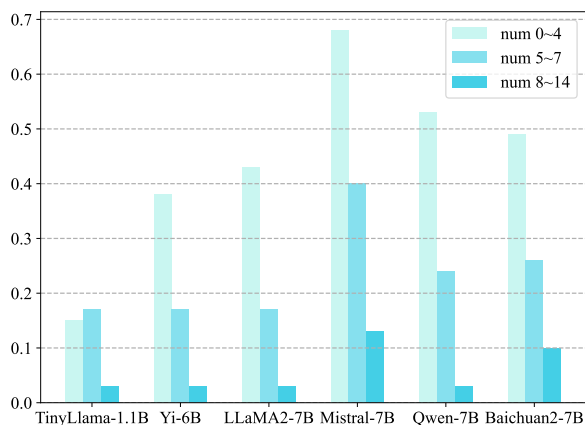


Figure 8: The scores of count & navigation task. We split the task by the statistics elements’ number. More elements indicate more need for attention points.

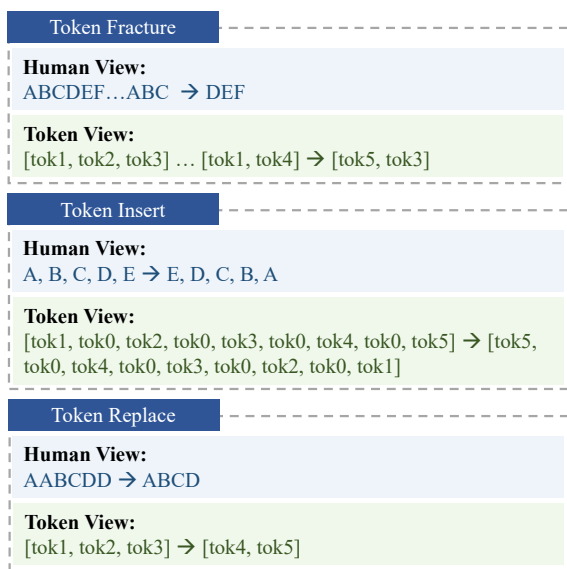


Figure 9: The three tokenization phenomena in which token views are different from human views.

fact that the special symbols that we can easily overlook also may require 1-3 tokens for encoding; "Token replace" indicates that several consecutive characters can be encoded as a single token, resulting in different encoded tokens for the reversed string compared to the original one.

4 Related Work

In-Context Learning The generalization ability of language models has always been a goal pursued by researchers, and the discovery of in-context learning ability is a milestone in achieving this goal. It was discovered in GPT-3 (Brown et al., 2020) that the model can implement custom outputs based on instructions and examples. In-context learning

can enable models to adapt to new tasks and even learn new knowledge and rules without updating model parameters. The phenomenon of in-context learning has attracted widespread attention from researchers to explore (Lu et al., 2021; Nie et al., 2022; Ye et al., 2022; Min et al., 2022; Liu et al., 2023a), explain (Xie et al., 2021; Akyürek et al., 2022; Han et al., 2023; Li et al., 2023), enhance (Min et al., 2021; Chen et al., 2021b; Yang et al., 2023), and apply (Dua et al., 2022; Shridhar et al., 2022; Wu et al., 2023) it. Chain-of-thought (CoT) reasoning is special in-context learning, divided into two modes: few-shot CoT (Wei et al., 2022) and zero-shot CoT (Kojima et al., 2022). Few-shot prompts is an important part of CoT, and many studies (Wang et al., 2022; Zhou et al., 2022; Fu et al., 2022; Lyu et al., 2023; Shum et al., 2023; Diao et al., 2023) focus on selecting better examples to promote the application of CoT. we propose an ICLEval benchmark for measuring in-context learning abilities. To the best of our knowledge, it’s the first work to estimate ICL abilities comprehensively.

Model Evaluation After the emergence of the large model, to comprehensively understand the boundaries and behaviors of its abilities, researchers tested the model from multiple aspects such as language ability, knowledge, professional ability, theory of mind, and security. The testing of model language abilities can utilize various traditional NLP task datasets, including language comprehension (Qin et al., 2023a), language generation (Qin et al., 2023a), and multilingualism (Lai et al., 2023). The testing of model knowledge includes common sense knowledge (Clark et al., 2018; Bisk et al., 2020; Zellers et al., 2019; Mihaylov et al., 2018), factual knowledge (Kwiatkowski et al., 2019; Lin et al., 2021; Berant et al., 2013), technical knowledge (Yuan et al., 2023), etc. The Achievement test for model specialty includes mathematics (Cobbe et al., 2021; Ling et al., 2017; Hendrycks et al., 2021), coding (Austin et al., 2021; Chen et al., 2021a), medicine (Thirunavukarasu et al., 2023), etc. In (Bubeck et al., 2023), theory of mind is considered to test GPT-4 (OpenAI, 2023). The test of model security includes stability, ethics, biases, and hallucination. Among them, many benchmarks can perform relatively comprehensive evaluation on the models, such as HELM (Liang et al., 2022), MMLU (Hendrycks et al., 2020), C_EVAL (Huang

et al., 2023), AGIEval(Zhong et al., 2023), BIG-Bench(Srivastava et al., 2022), etc. However, most of the evaluations here need to involve a variety of different basic capabilities, and the quality of the test results is affected by a variety of factors. Our ICLEval benchmark decouples the evaluation of the in-context learning ability from other abilities.

5 Conclusion

In this work, we introduce an ICLEval benchmark for measuring the in-context learning ability of LLMs. We classify it into copying ability and learning ability, and design 12 evaluation tasks. We find that model size is an important but not the sole determinant of ICL ability. Also, we observe that ICL abilities, particularly copying ability have a quick increase in the very early pretraining stage. Furthermore, we analyze some bad cases and discover that the ICL abilities are influenced by distinguishing ability, inherent preferences, attention points capacity, and tokenizer. That indicates that we need to consider these factors if we would like to train a model with strong in-context ability.

Limitations

When designing the ICLEval evaluation, we minimize the need for other abilities of the model as much as possible. However, we still can't guarantee that the testing tasks will not be affected by the language abilities and internal knowledge of the model itself. For example, the inherent preferences we discussed in Section 3.4 might make LLMs ignore the context.

Acknowledgements

We sincerely thank all the anonymous reviewers and (S)ACs for their constructive comments and helpful suggestions. This work was supported by The National Natural Science Foundation of China (No. 62376273).

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Baichuan. 2023. *Baichuan 2: Open large-scale language models*. *arXiv preprint arXiv:2309.10305*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

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

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

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. 2021a. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2021b. Meta-learning via language model in-context tuning. *arXiv preprint arXiv:2110.07814*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. *arXiv preprint arXiv:2212.04092*.

- Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.
- Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. 2023. In-context learning of large language models explained as kernel regression. *arXiv preprint arXiv:2305.12766*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.
- Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, et al. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *arXiv preprint arXiv:2305.08322*.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada. Association for Computational Linguistics.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.
- Viet Dac Lai, Nghia Trung Ngo, Amir Pouran Ben Veyseh, Hieu Man, Franck Dernoncourt, Trung Bui, and Thien Huu Nguyen. 2023. Chatgpt beyond english: Towards a comprehensive evaluation of large language models in multilingual learning. *arXiv preprint arXiv:2304.05613*.
- Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Pappalopoulos, and Samet Oymak. 2023. Transformers as algorithms: Generalization and stability in in-context learning.
- Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.
- Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.
- Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.
- Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, Richard Fan, Yi Gu, Victor Miller, Yonghao Zhuang, Guowei He, Haonan Li, Fajri Koto, Liping Tang, Nikhil Ranjan, Zhiqiang Shen, Xuguang Ren, Roberto Iriondo, Cun Mu, Zhiting Hu, Mark Schulze, Preslav Nakov, Tim Baldwin, and Eric P. Xing. 2023b. [Llm360: Towards fully transparent open-source llms](#).
- Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.
- Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.
- Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.
- Sewon Min, Xinxin Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.
- Feng Nie, Meixi Chen, Zhirui Zhang, and Xu Cheng. 2022. Improving few-shot performance of language models via nearest neighbor calibration. *arXiv preprint arXiv:2212.02216*.
- Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

- OpenAI. 2023. Gpt-4 technical report.
- Tianduo Wang Peiyuan Zhang, Guangtao Zeng and Wei Lu. 2023. *Tinyllama*.
- Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023a. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.
- Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, et al. 2023b. Tool learning with foundation models. *arXiv preprint arXiv:2304.08354*.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.
- Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.
- Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2022. Distilling multi-step reasoning capabilities of large language models into smaller models via semantic decompositions. *arXiv preprint arXiv:2212.00193*.
- KaShun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. *arXiv preprint arXiv:2302.12822*.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.
- InternLM Team. 2023. Internlm: A multilingual language model with progressively enhanced capabilities. <https://github.com/InternLM/InternLM>.
- Arun James Thirunavukarasu, Refaat Hassan, Shathar Mahmood, Rohan Sanghera, Kara Barzangi, Mohammed El Mukashfi, and Sachin Shah. 2023. Trialling a large language model (chatgpt) in general practice with the applied knowledge test: observational study demonstrating opportunities and limitations in primary care. *JMIR Medical Education*, 9(1):e46599.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shrutu Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.
- Zhenyu Wu, YaoXiang Wang, Jiacheng Ye, Jiangtao Feng, Jingjing Xu, Yu Qiao, and Zhiyong Wu. 2023. Openicl: An open-source framework for in-context learning. *arXiv preprint arXiv:2303.02913*.
- Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.
- Jiaxi Yang, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. 2023. Iterative forward tuning boosts in-context learning in language models. *arXiv preprint arXiv:2305.13016*.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.
- Xi Ye, Srinivasan Iyer, Asli Celikyilmaz, Ves Stoyanov, Greg Durrett, and Ramakanth Pasunuru. 2022. Complementary explanations for effective in-context learning. *arXiv preprint arXiv:2211.13892*.
- Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, and Songfang Huang. 2023. How well do large language models perform in arithmetic tasks? *arXiv preprint arXiv:2304.02015*.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.
- Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A ICLEval Benchmark

A.1 Data Source

The data for our ICLEval Benchmark is sourced from various places, including Wikipedia, a common noun vocabulary, GSM8K, AQuA, BIGBench, and generation context from ChatGPT.

For paragraph-level and sentence-level data, we collected information from Wikipedia. Initially, we randomly selected 100 articles from Wikipedia to form the basis of natural language paragraphs. These paragraphs serve as the core content for the string completion task, which tests the models' ability to accurately copy information in unstructured context. It's important to note that we processed these paragraphs to ensure that the models hadn't encountered them during the pre-training stage. Additionally, we extracted all the sentences containing 5-30 words from these Wikipedia paragraphs to create sentence-level data. This data is used for tasks such as dictionary search, order adjustment, and de-duplication.

For word-level data, we gathered 1526 common nouns from a common noun vocabulary. Furthermore, we included all numbers, uppercase letters, and lowercase letters as character-level data. These datasets primarily focus on tasks related to solving order-related problems.

In addition, we sampled 50 examples from the GSM8K and AuQA datasets, respectively, to create the format cloning task. It's important to note that we are not concerned with the accuracy of the math questions themselves, but rather with the format of the predictions. Furthermore, we generated virtual data in the "person" and "company" domains using ChatGPT, which serves as the foundational content for tasks such as format check and format conversion.

For tasks related to statistics problems, we drew inspiration from the "navigate" and "long_context_integration" tasks in BIG-Bench (Srivastava et al., 2022). Additionally, we utilized the data from the "list_numbers" task to create our numbers' rules task.

A.2 Tasks

We partition our tasks into two categories: testing copying abilities and testing learning abilities. The first two tasks focus on assessing copying abilities, while the remaining tasks evaluate learning abilities.

String Completion task is specifically designed to test copying abilities in natural language context scenarios. In this task, models are required to predict the second half of a string given the first half. For each testing sample, we randomly select a lengthy paragraph from Wikipedia as the basis for a natural language context. To ensure that the model has not encountered the main entity mentioned in the paragraph before, we replace it with a 16-character hash string. Furthermore, we split the last hash string in the paragraph into two 8-character halves, and the target for the model is to predict the subsequent 8 characters. We have designed this task because it closely resembles the pre-training task and serves as a fundamental form of assessing copying ability. An example of this task can be seen in Figure 16.

Dictionary Search task involves predicting a value based on a given key from a set of key-value pairs. We explore two different forms of this task. The first form consists of short keys and long values, while the second form features short values but longer, more intricate keys. The samples in this task can be seen in Figure 16.

In the first form, we provide 20 key-value pairs with varying lengths as in-context examples. The key is a 6-character random hash string, while the value is a random sentence sampled from Wikipedia.

The second form might be more difficult. We provide 10 key-value pairs that involve number calculations as in-context examples. The key is a long number calculation string with more than 20 numbers and operators. Moreover, to examine the impact of similar keys, we establish three levels for this task: "all-similar," "half-similar," and "non-similar." These levels determine the number of keys that share a similarity with the prediction target.

Format Check is a classification task to distinguish which class the current format is. We chose six common formats: JSONL, CSV, Triple-Tuple, YAML, Markdown-Table, and XML. Then we will randomly generate 6-shot examples using the six formats respectively as the in-context for each testing sample. Each sample is a random format and

belongs to the six formats. The samples in this task can be seen in Figure 17.

Format Cloning is a generation task to follow the customized formats. We have designed five customized formats for the CoT output of GSM8K (Cobbe et al., 2021) and the multi-choice output of AQuA (Ling et al., 2017). And we randomly sample 5-shot examples from this dataset. In these tasks, we check the correctness of the output format without focusing on the correctness of the results themselves. The samples in this task can be seen in Figure 17.

Format Conversion is a generation task to convert a source format to a target format and keep the content consistent. We also use the six formats mentioned in the "format-check" task. To explore more complex scenarios, we set four forms in this task: "single", "multi", "transfer" and "mix". We have two domains ("people" and "company") of data as content. "single" form means only converting one item, and the domain is consistent between in-context examples and testing samples. "multi" form will have 1-5 items based on the "single" form. While the "transfer" form also converts one item the in-context examples and testing samples are from different domains. The "mix" form might most difficult which means there will be multiple items and both two domains' data as content. The samples in this task can be seen in Figure 18.

Order Check is a classification task to judge if the order of elements in one string is reversed. We set 8-shot examples and each example has two strings. If the two strings have the same order, the label is False, while if the two strings have reversed order, the label is True. We will randomly generate different 8-shot examples for every testing sample, and the proportion of positive and negative examples is also random. We have word-level and character-level settings for different element granularity. The sample in this task can be seen in Figure 19.

Order Adjustment is a generation task to generate sequence, reverse, or specify form string given origin string. For the sequence setting, the model is only required to copy the input text without any changes. For the reverse sequence setting, the model needs to output the reversed version of the original text. For the specific sequence setting, the model must learn to identify and execute a specified index operation on the input text. We randomly generate 5-shot examples for each testing

sample, and hope models can learn and apply the order adjustment rules to the testing sample. To enhance diversity, we experiment with character-level, word-level, and sentence-level input-output pairs for different element granularity. The sample in this task can be seen in Figure 19.

Count & Navigation is a generation task to return a dictionary depending on the in-context information. The model is given a list of dictionary nouns (up, down, right, and left) as input. The count task requires the model to count the number of occurrences of all the nouns, while the navigation task involves considering the final point relative to the initial point. We set "easy" mode and "hard" mode for this task. The "easy" mode only has two dictionary nouns while the "hard" mode has four. The sample in this task can be seen in Figure 20.

Relation Analysis task provides the model with a relation graph as input and requires it to output all the nodes connected to a chosen node. Each of these tasks includes 5-shot examples as in-context information. We have adjusted the nodes or sides in the relation graph for this task, to introduce variations for different models. The sample in this task can be seen in Figure 20.

Duplication Check a classification task to judge if there are repeated elements in the string. If there are repeated elements in the string, the label is True, else the label is False. We use the same setting as the "order-check" task, which has 8-shot examples for every testing sample and has word-level and character-level element granularity. The sample in this task can be seen in Figure 20.

De-Duplication is a generation task to find or remove repeat elements from a string. Finding repeat elements means only outputting the repeat element in one string while removing repeat elements means outputting a whole string without any repeat elements. We use sentence-level settings for finding repeat elements form and use character-level and word-level settings for removing repeat elements form. As previous setting, we randomly generate 5-shot examples for each testing sample. The sample in this task can be seen in Figure 20.

Numbers' List Mapping is a task that consists of multiple groups of number pairs. Each pair comprises an input list and an output list, with an internal rule governing the transformation from input to output. With multi-group examples as the in-context information, the model needs to learn the

underlying rule and predict the output list for a new input list. We have collected task data for this evaluation from the "list_functions" task in BIGBench. In this task, there are 250 different rules, and some rules are even very hard for humans to find. The samples in this task can be seen in Figure 15.

Task Name	Split String	Max Len.
Copying		
entity complete	all punctuation marks	10
dictionary search		
- short key	\n	75
- long and similar key	\n	12
Learning		
format check	\n	5
format cloning	\nQuestion:	196
format conversion	\nInput:	256
order check	\n	5
order adjustment		
- character-level	\n	50
- word-level	\n	50
- sentence-level	\nInput:	256
count & navigation	\n	30
relation analysis	\nInput:	60
duplication check	\n	5
de-duplication		
- character-level	\n	30
- word-level	\n	30
- sentence-level	\n	60
numbers' rules	\n	50

Table 4: The max generation length and split string for different tasks. We set the max generation length for testing effectiveness.

A.3 Metrics

Considering the effectiveness of ICLEval, we set different max generation length limitations for various tasks. Meanwhile, we set different split strings for processing models' responses to final predictions. Both the settings are shown in Table 4. Setting the two parameters is because the base-version model will not stop when giving in-context examples. The max generation length will make tested models stop their generation in the suit position. And we will regard the content in front of the split strings as the right predictions. Then we will strip the blank space at both ends of the right predictions.

B Evaluated Models

We show the details of our evaluated models in this section, and the models are listed in Table 5.

GPT-Neo-1.3B is a transformer model with 1.3 billion parameters, trained on the Pile, a curated dataset created by EleutherAI specifically for training this model.

Model	Version	Parameters	Training Data
GPT-Neo	Base	1.3B	0.38T Tokens
Phi-1.5	Base	1.3B	-
TinyLlama	Base	1.1B	3T Tokens
GPT-J	Base	6B	0.4T Tokens
		7B	1T Tokens
LLaMA1	Base	13B	1T Tokens
		34B	1.4T Tokens
		65B	1.4T Tokens
LLaMA2	Base/Chat	7B	2T Tokens
		13B	2T Tokens
LLaMA3	Base	70B	2T Tokens
		8B	>15T Tokens
		70B	-
Mistral	Base/Chat	7B	-
Mistral-MoE	Chat	8×7B	-
ChatGLM3	Base/Chat	6B	-
YI	Base/Chat	6B	3T Tokens
		34B	3T Tokens
Baichuan2	Base/Chat	7B	2.6T Tokens
		13B	2.6T Tokens
Qwen	Base/Chat	7B	>2.4T Tokens
		14B	>3T Tokens
InternLM	Base/Chat	7B	>1T Tokens
Skywork	Base	13B	3.2T Tokens
Amber	Base	7B	1.2T Tokens

Table 5: Models evaluated in our work

Phi-1.5 is a transformer model with 1.3 billion parameters, trained by Microsoft Research. It is specialized in basic Python coding and is augmented with various NLP synthetic texts.

TinyLlama project (Peiyuan Zhang and Lu, 2023) aims to pretrain a 1.1 billion parameter Llama model on 3 trillion tokens. The model's intermediate checkpoints, corresponding to about 10 billion tokens, are open-sourced every 5,000 steps. This project is released by the StatNLP Research Group of Singapore University of Technology and Design.

GPT-J is a transformer model trained with 6 billion parameters. It is trained by EleutherAI.

ChatGLM3-6B is the latest open-source model in the ChatGLM series, utilizing the General Language Model (GLM) architecture with 6 billion parameters. It was released by the Knowledge Engineering Group (KEG) & Data Mining at Tsinghua University.

YI-6B and YI-34B belong to the Yi series models, trained from scratch by 01.AI. These models are based on a 3 trillion multilingual corpus and have 6 billion and 34 billion parameters, respectively.

LLaMA1-7B and LLaMA1-13B belong to the Llama series models released by Meta AI. They are

trained on 1 trillion tokens.

LLaMA1-30B and LLaMA1-65B belong to the Llama series models released by Meta AI. They are trained on 1.4 trillion tokens.

LLaMA2-7B, LLaMA2-13B, and LLaMA2-70B belong to the Llama 2 series models, which are a collection of pre-trained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. These models are autoregressive language models trained on 2 trillion tokens and were released by Meta AI.

Mistral-7B-v0.1 is a pre-trained generative text model with 7 billion parameters. It was released by Mistral AI.

Skywork-13B was trained on a high-quality cleaned dataset consisting of 3.2 trillion multilingual data, mainly in Chinese and English, including code. It was released by Skywork AI.

Baichuan2-7B and Baichuan2-13B belong to the Baichuan2 series models, which are open-source and commercially usable large-scale language models developed by Baichuan Intelligence. They are trained on a high-quality corpus with 2.6 trillion tokens. Baichuan Intelligence also opensources intermediate checkpoints of Baichuan2-7B every 220 billion tokens.

Qwen-7B and Qwen-14B are versions of the large language model series called Qwen (Tongyi Qianwen) proposed by Alibaba Cloud. These Transformer-based large language models have 7 billion, 14 billion, and 72 billion parameters, respectively. They are trained on a large volume of data, including web texts, books, code, etc.

InternLM-7B (Team, 2023) is an open-sourced base model with 7 billion parameters, tailored for practical scenarios.

Amber-7B is a 7 billion parameter language model with the same architecture as LLaMA-7B, trained on Arxiv, Book, C4, Refined-Web, StarCoder, StackExchange, and Wikipedia. It was released by LLM360 (Liu et al., 2023b), which is an initiative for comprehensive and fully open-source language models. And it provides open-source 360 intermediate checkpoints of Amber-7B.

C Experiments Results

C.1 Bad Cases Analysis

We list some bad cases in this sub-section.

Many models can arrive at about 0.89 scores in the entity completion task, but it’s hard for them to get higher scores. We analyzed the bad case and

Copying (String Completion)	
Prompt:	The Baptist Mission Union censured 521351caba13f591 for abandoning his post without authorization and going to Rangoon; ... Calista carried on the mission work, assisted by her son Brainerd and daughter Calista and their spouses, who returned from America after their studies. In 1872, long after the death of 521351ca
“521351caba13f591” to tokens:	[29871, 29945, 29906, 29896, 29941, 29945, 29896, 29883, 5363, 29896, 29941, 29888, 29945, 29929, 29896]
“521351ca” to tokens:	[29871, 29945, 29906, 29896, 29941, 29945, 29896, 1113]
LLaMA2-7B:	13f591
Mistral-7B:	13f591
Yi-6B:	ab13f591
Baichuan2-7B:	13f591
Qwen-7B:	13f591
Yi-34B:	13f591
LLaMA-65B:	13f591
Label:	ba13f591
Metrics:	we will check if the prediction hash string is exact match with the label hash string.

Figure 10: A bad case of entity completion task and order adjustment task. These two bad cases are selected from the predictions of LLaMA2-7B.

found that there are some entities (hash string) that have token-fracture phenomena when splitting them. As shown in Figure 10, the tokens of string “521351caba13f591” are not equal to the tokens of “521351ca” add the tokens of “ba13f591”. This makes it difficult for models to finish this task.

We give a bad case of dictionary search task in Figure 12. Similar examples confuse the models, and only the LLaMA-65B model finds the right value.

We give a bad case of format check task in Figure 13. Lots of models predict the results depending on the models’ inherent preferences rather than the in-context examples. The rules in this task are easy to learn so that small-sized models (e.g. TinyLlama) can predict the right labels. However, the larger models (e.g. LLaMA-65B) might predict wrong labels due to their inherent preferences.

We give two bad cases of count & navigation task in Figure 14. Models have good performance when there are several elements to statistics. The performance has a huge drop when the number of elements becomes greater.

We give two bad cases of numbers’ rules task in

N-shot	LLaMA2-7B		LLaMA2-13B		Mean
	normal	transfer	normal	transfer	
6-shot	0.017	0.300	0.000	0.233	0.138
12-shot	0.983	1.000	0.983	0.967	0.983
24-shot	1.000	0.983	1.000	1.000	0.996

Table 6: Different n-shot settings in Format Checking Task. "normal" and "transfer" are two different subsets in this task.

Figure 15. Some rules are even very difficult for humans. We chose two hard samples and showed the mean human score in this figure.

C.2 Detailed Results

We list the detailed results in Table 7. Each column is the results of all the models we tested for one task. We list the mean results of all the tasks in the last column of this table. We also put the maximum and minimum results in the last two rows of the table.

C.3 Results in Pretraining Stage

In this section, we tested the Amber-7B models with multiple checkpoints, trained on 1.2T tokens, and saved about every 3.5B tokens. The ICL abilities scores are shown in Figure 11. We can also draw similar conclusions to the results of Baichuan2-7B and TinyLlama-1.1B.

C.4 Influences of Evaluation Settings.

Different evaluation settings might have can affect the results of ICL abilities, such as the order, the prompt, and the number of examples.

(1) We dynamically sample every example for each individual sample within our tasks, as opposed to using a fixed set of examples for all samples. Therefore, we believe that settings effectively address the issue of order.

(2) We only use the "input-output" as the prefix for all the examples in our main experiment. We try some experiments to set different prompts, such as "Question-Answer", "123-456", "Q-A", and "answer-question" to replace our original prompt "Input-Output", as shown in Table 8. The results are shown in the table below. We can see that 1) the prompt with influence the models' results when the number of examples is not enough, but has little influence when the number of examples rises. 2) the prompt settings nearly do not influence the relative results between different models. So, there is reason to believe that the prompt setting will not

influence the conclusion of our main experiment. In considering the robust evaluation, we will add different prompts in our testing data in the future.

(3) More examples will bring better results. We select the suitable n-shot settings for greater discrimination in our main experiment. We also adjusted the n-shot settings for several tasks (format checking and count & navigation), and the results are shown in Table 6 and Table 9.

Models	String Comp.	Dict. Search	Order Check	Order Adjust.	Dupl. Check	De-Dupl.	Rel. Anal.	Count & Nav.	Format Check	Format Cloning	Format Conv.	List Map.	Mean
Random	0.000	0.000	0.500	0.000	0.500	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.072
TinyLlama-1.1B	0.458	0.890	0.540	0.379	0.557	0.223	0.010	0.125	0.375	0.827	0.592	0.356	0.498
Phi-1.5	0.253	0.540	0.550	0.379	0.573	0.287	0.030	0.100	0.175	0.720	0.550	0.388	0.396
GPT-Neo-1.3B	0.479	0.540	0.610	0.387	0.533	0.180	0.010	0.092	0.175	0.777	0.492	0.376	0.430
GPT-J	0.463	0.530	0.730	0.438	0.637	0.233	0.030	0.133	0.617	0.833	0.675	0.492	0.506
ChatGLM3-6B	0.700	0.900	0.750	0.333	0.517	0.273	0.170	0.317	0.483	0.840	0.475	0.432	0.582
ChatGLM3-6B-Chat	0.342	0.890	0.610	0.363	0.550	0.327	0.100	0.225	0.675	0.887	0.725	0.372	0.526
Yi-6B	0.379	0.870	0.740	0.488	0.500	0.340	0.070	0.217	0.100	0.943	0.792	0.492	0.541
Yi-6B-Chat	0.484	0.870	0.640	0.525	0.623	0.447	0.140	0.258	0.033	0.863	0.808	0.452	0.554
Yi-34B	0.600	0.860	0.980	0.696	0.730	0.627	0.470	0.525	0.117	0.960	0.808	0.536	0.675
Skywork-13B	0.458	0.880	0.930	0.438	0.657	0.357	0.130	0.350	0.492	0.953	0.808	0.472	0.603
InternLM-7B	0.421	0.550	0.630	0.404	0.577	0.277	0.100	0.158	0.317	0.877	0.767	0.456	0.479
InternLM-7B-Chat	0.732	0.530	0.640	0.308	0.620	0.000	0.000	0.258	0.575	0.427	0.425	0.380	0.468
Qwen-7B	0.453	0.890	0.750	0.487	0.687	0.523	0.150	0.333	0.100	0.927	0.700	0.484	0.574
Qwen-7B-Chat	0.721	0.890	0.630	0.354	0.643	0.337	0.110	0.300	0.083	0.813	0.342	0.492	0.559
Qwen-14B	0.484	0.870	0.870	0.717	0.707	0.637	0.350	0.467	0.308	0.923	0.808	0.552	0.653
Qwen-14B-Chat	0.584	0.900	0.770	0.683	0.720	0.623	0.350	0.467	0.167	0.546	0.758	0.552	0.632
Baichuan2-7B	0.442	0.870	0.610	0.454	0.603	0.337	0.110	0.308	0.158	0.933	0.800	0.492	0.551
Baichuan2-7B-Chat	0.516	0.880	0.670	0.429	0.557	0.403	0.140	0.233	0.058	0.773	0.642	0.456	0.538
Baichuan2-13B	0.232	0.800	0.870	0.450	0.617	0.467	0.210	0.325	0.633	0.893	0.725	0.392	0.540
Baichuan2-13B-Chat	0.363	0.750	0.830	0.396	0.630	0.510	0.270	0.392	0.575	0.887	0.683	0.356	0.541
Mistral-7B	0.537	0.880	0.980	0.575	0.623	0.490	0.300	0.442	0.217	0.947	0.842	0.560	0.648
Mistral-7B-Chat	0.763	0.870	0.720	0.471	0.603	0.493	0.190	0.358	0.450	0.917	0.833	0.468	0.640
Mixtral-8x7B-Chat	0.884	0.880	0.990	0.737	0.793	0.677	0.530	0.517	0.358	0.953	0.833	0.592	0.761
LLaMA-7B	0.332	0.870	0.670	0.412	0.587	0.327	0.030	0.150	0.358	0.900	0.775	0.416	0.518
LLaMA-13B	0.426	0.890	0.650	0.442	0.577	0.340	0.080	0.275	0.192	0.907	0.800	0.472	0.547
LLaMA-30B	0.332	0.890	0.910	0.629	0.637	0.447	0.350	0.425	0.475	0.940	0.783	0.528	0.619
LLaMA-65B	0.495	0.890	0.970	0.729	0.613	0.563	0.330	0.392	0.000	0.947	0.833	0.576	0.646
LLaMA2-7B	0.453	0.890	0.830	0.429	0.630	0.293	0.120	0.233	0.158	0.917	0.767	0.452	0.560
LLaMA2-7B-Chat	0.758	0.890	0.730	0.454	0.623	0.403	0.220	0.317	0.258	0.920	0.650	0.412	0.609
LLaMA2-13B	0.474	0.890	0.830	0.608	0.620	0.423	0.090	0.308	0.117	0.947	0.750	0.520	0.595
LLaMA2-13B-Chat	0.747	0.900	0.750	0.604	0.620	0.467	0.260	0.292	0.133	0.900	0.658	0.488	0.631
LLaMA2-70B	0.432	0.890	0.980	0.758	0.747	0.687	0.470	0.450	0.483	0.933	0.842	0.580	0.685
LLaMA3-8B	0.874	0.570	0.980	0.892	0.670	0.700	0.560	0.517	0.300	0.947	0.833	0.628	0.719
LLaMA3-70B	0.911	0.640	0.990	0.883	0.853	0.977	0.970	0.675	0.867	0.953	0.867	0.660	0.819
Maximum	0.911	0.900	0.990	0.892	0.853	0.977	0.970	0.675	0.867	0.960	0.867	0.660	0.819
Minimum	0.232	0.530	0.540	0.308	0.500	0.000	0.000	0.092	0.000	0.427	0.342	0.356	0.396

Table 7: The detailed results for ICLEval of all evaluated models. The complete task names are: Dictionary Search, String Completion, Order Check, Order Adjustment, Duplication Check, De-Duplication, Relation Analysis, Count and Navigation, Format Check, Format Cloning, Format Conversion, and List Mapping.

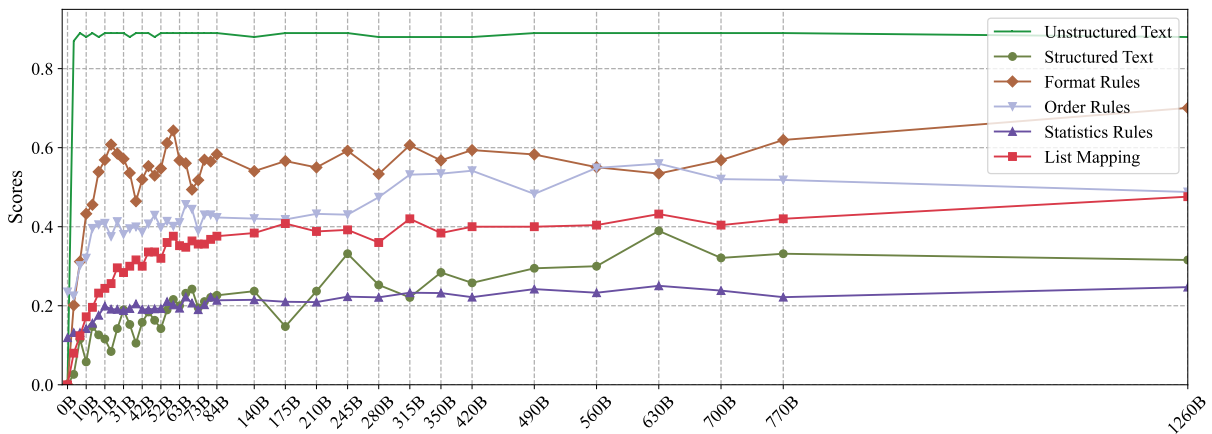


Figure 11: The pretrain stage of Amber-7B.

Prompt Type	Input-Output		Question-Answer		123-456		Q-A		answer-question	
	normal	transfer	normal	transfer	normal	transfer	normal	transfer	normal	transfer
LLaMA2-7B (6-shot)	0.02	0.30	0.23	0.20	0.28	0.23	0.30	0.27	0.18	0.13
LLaMA2-13B (6-shot)	0.00	0.23	0.42	0.23	0.38	0.23	0.40	0.27	0.28	0.20
LLaMA2-7B (24-shot)	1.00	0.98	1.00	0.98	1.00	0.98	1.00	1.00	1.00	0.97
LLaMA2-13B (24-shot)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.97	0.97

Table 8: Different prompts and n-shot settings in Format Checking Task. "normal" and "transfer" are two different subsets in this task.

N-shot	LLaMA2-7B				LLaMA2-13B				Mean
	count-1dim	count-2dim	nav.-1dim	nav.-2dim	count-1dim	count-2dim	nav.-1dim	nav.-2dim	
4-shot	0.367	0.067	0.233	0.033	0.500	0.067	0.267	0.033	0.196
8-shot	0.467	0.067	0.267	0.133	0.567	0.200	0.400	0.067	0.271
16-shot	0.567	0.000	0.467	0.067	0.833	0.100	0.567	0.033	0.329

Table 9: Different n-shot settings in Count & Navigation Task. "count-1dim", "count-2dim", "navigation-1dim", and "navigation-2dim" are four different subsets in this task.

Copying (Dictionary Search)

Prompt:
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -35939
73 ∩ 5 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -35894
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -36439
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 76 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 145 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -36662
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 137 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -35936
73 ∩ 55 * 98 ∩ 858 * 824 * 748 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -36002
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 152 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -7739
72 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -35940
73 ∩ 55 * 8 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -36029
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠 -35939
73 ∩ 55 * 98 ∩ 858 * 894 * 741 * 51 * 86 ☆ 72 ∩ 44 ∩ 758 ∩ 23 * 148 * 348 ∩ 159 * 134 ∩ 452 ☆ 94 * 891 ∩ 88 ∩ 526 * 97 ∩ 607 🏠

LLaMA2-7B:	-36029	Yi-34B:	-35940
Mistral-7B:	-35940	LLaMA-65B:	-35939
Yi-6B:	-35941	Label:	-35939
Baichuan2-7B:	-36002	Metrics: we will check if the prediction number is exact match with the label number.	
Qwen-7B:	-36001		

Figure 12: A bad case of dictionary search task (with all-similar examples).

Learning (Format Check)

<p>Prompt: Input: Index name age city --- --- --- --- 1 Ava Hill 31 Portland Output: table</p> <p>Input: Index,name,age,city 1,David Wilson,29,Boston Output: csv</p> <p>Input: (Landon Smith, age, 31) (Landon Smith, city, New Orleans) Output: tuple</p> <p>Input: <person> <name>Alexander Harris</name> <age>33</age> <city>Philadelphia</city> </person> Output: xml</p> <p>Input: person: name: Charlotte Adams age: 35 city: Seattle</p>	<p>Output: yaml</p> <p>Input: {"name": "Grace Morgan", "age": 30, "city": "St. Louis"} Output: jsonl</p> <p>Input: Index name age city --- --- --- --- 1 Brooklyn Wilson 35 Tampa Output:</p> <table border="1"> <tr><td>Tiny Llama:</td><td>table</td></tr> <tr><td>LLaMA2-7B:</td><td>csv</td></tr> <tr><td>Mistral-7B:</td><td>table</td></tr> <tr><td>Yi-6B:</td><td>csv</td></tr> <tr><td>Baichuan2-7B:</td><td>csv</td></tr> <tr><td>Qwen-7B:</td><td>csvl</td></tr> <tr><td>Yi-34B:</td><td>json</td></tr> <tr><td>LLaMA-65B:</td><td>json</td></tr> <tr><td>Label:</td><td>table</td></tr> </table> <p>Metrics: we check if the prediction results exact match with the labels.</p>	Tiny Llama:	table	LLaMA2-7B:	csv	Mistral-7B:	table	Yi-6B:	csv	Baichuan2-7B:	csv	Qwen-7B:	csvl	Yi-34B:	json	LLaMA-65B:	json	Label:	table
Tiny Llama:	table																		
LLaMA2-7B:	csv																		
Mistral-7B:	table																		
Yi-6B:	csv																		
Baichuan2-7B:	csv																		
Qwen-7B:	csvl																		
Yi-34B:	json																		
LLaMA-65B:	json																		
Label:	table																		

Figure 13: A bad case of format check task.

Learning (Count & Navigation)

<p>Prompt: ... (n-shot examples)</p> <p>Input: down, down Output: up: 0, down: 2</p> <p>Input: up Output: up: 1, down: 0</p> <p>Input: up, up, down Output:</p> <table border="1"> <tr><td>Tiny Llama:</td><td>up: 2, down: 1</td></tr> <tr><td>LLaMA2-7B:</td><td>up: 2, down: 1</td></tr> <tr><td>Mistral-7B:</td><td>up: 2, down: 1</td></tr> <tr><td>Yi-6B:</td><td>up: 2, down: 1</td></tr> <tr><td>Baichuan2-7B:</td><td>up: 2, down: 1</td></tr> <tr><td>Qwen-7B:</td><td>up: 2, down: 1</td></tr> <tr><td>Yi-34B:</td><td>up: 2, down: 1</td></tr> <tr><td>LLaMA-65B:</td><td>up: 2, down: 1</td></tr> <tr><td>Label:</td><td>up: 2, down: 1</td></tr> </table> <p>Metrics: we will check if the prediction dictionary string is exact match with the label dictionary string.</p>	Tiny Llama:	up: 2, down: 1	LLaMA2-7B:	up: 2, down: 1	Mistral-7B:	up: 2, down: 1	Yi-6B:	up: 2, down: 1	Baichuan2-7B:	up: 2, down: 1	Qwen-7B:	up: 2, down: 1	Yi-34B:	up: 2, down: 1	LLaMA-65B:	up: 2, down: 1	Label:	up: 2, down: 1	<p>Prompt: ... (n-shot examples)</p> <p>Input: left, left, right, left, down, left, right, right Output: up: 0, down: 1, left: 4, right: 3</p> <p>Input: up, down, up, right, right, right, right, down Output: up: 2, down: 2, left: 0, right: 4</p> <p>Input: left, right, down, down, right, right, left, up, up, up Output:</p> <table border="1"> <tr><td>Tiny Llama:</td><td>up: 3, down: 3, left: 3, right: 3</td></tr> <tr><td>LLaMA2-7B:</td><td>up: 0, down: 2, left: 2, right: 4</td></tr> <tr><td>Mistral-7B:</td><td>up: 3, down: 2, left: 3, right: 3</td></tr> <tr><td>Yi-6B:</td><td>up: 3, down: 3, left: 3, right: 3</td></tr> <tr><td>Baichuan2-7B:</td><td>up: 4, down: 4, left: 4, right: 4</td></tr> <tr><td>Qwen-7B:</td><td>up: 3, down: 3, left: 3, right: 3</td></tr> <tr><td>Yi-34B:</td><td>up: 3, down: 2, left: 2, right: 3</td></tr> <tr><td>LLaMA-65B:</td><td>up: 3, down: 3, left: 3, right: 3</td></tr> <tr><td>Label:</td><td>up: 3, down: 2, left: 2, right: 3</td></tr> </table>	Tiny Llama:	up: 3, down: 3, left: 3, right: 3	LLaMA2-7B:	up: 0, down: 2, left: 2, right: 4	Mistral-7B:	up: 3, down: 2, left: 3, right: 3	Yi-6B:	up: 3, down: 3, left: 3, right: 3	Baichuan2-7B:	up: 4, down: 4, left: 4, right: 4	Qwen-7B:	up: 3, down: 3, left: 3, right: 3	Yi-34B:	up: 3, down: 2, left: 2, right: 3	LLaMA-65B:	up: 3, down: 3, left: 3, right: 3	Label:	up: 3, down: 2, left: 2, right: 3
Tiny Llama:	up: 2, down: 1																																				
LLaMA2-7B:	up: 2, down: 1																																				
Mistral-7B:	up: 2, down: 1																																				
Yi-6B:	up: 2, down: 1																																				
Baichuan2-7B:	up: 2, down: 1																																				
Qwen-7B:	up: 2, down: 1																																				
Yi-34B:	up: 2, down: 1																																				
LLaMA-65B:	up: 2, down: 1																																				
Label:	up: 2, down: 1																																				
Tiny Llama:	up: 3, down: 3, left: 3, right: 3																																				
LLaMA2-7B:	up: 0, down: 2, left: 2, right: 4																																				
Mistral-7B:	up: 3, down: 2, left: 3, right: 3																																				
Yi-6B:	up: 3, down: 3, left: 3, right: 3																																				
Baichuan2-7B:	up: 4, down: 4, left: 4, right: 4																																				
Qwen-7B:	up: 3, down: 3, left: 3, right: 3																																				
Yi-34B:	up: 3, down: 2, left: 2, right: 3																																				
LLaMA-65B:	up: 3, down: 3, left: 3, right: 3																																				
Label:	up: 3, down: 2, left: 2, right: 3																																				

Figure 14: The bad cases of Count & Navigation task.

Learning (Numbers' Rules)

Learning (Numbers' Rules)	
<p>Prompt: Input: [3, 41, 7, 1, 11, 77, 9, 61, 4, 17] Output: [41, 1, 77, 61, 17]</p> <p>Input: [36, 9, 1, 17, 29, 4, 93, 89] Output: [9, 17, 89]</p> <p>Input: [51, 61, 0, 9, 89, 85] Output: [61, 9, 85]</p> <p>Input: [83, 25, 95, 55, 46, 6, 2, 9] Output: [25, 55, 9]</p> <p>Input: [3, 5, 6, 21, 15, 77, 8, 86, 1] Output: [5, 21, 77]</p> <p>Input: [90, 31, 8, 2, 7, 24] Output: [31]</p> <p>Input: [61, 94, 4, 3, 65, 9, 30, 11, 6, 7] Output: [3, 9, 11, 7]</p> <p>Input: [9, 6, 2, 64, 15, 84] Output: []</p> <p>Input: [2, 17, 5, 62, 4, 99, 0] Output: [17, 99]</p> <p>Input: [8, 2, 27, 7, 37, 4, 36, 3, 62, 15] Output: [7, 3, 15]</p> <p>Input: [9, 8, 0, 58, 21, 3] Output: [3]</p> <p>Input: [6, 5, 0, 37, 52, 36] Output: [5, 37]</p> <p>Input: [4, 9, 6, 1, 8, 16, 97] Output: [9, 1]</p> <p>Input: [5, 1, 48, 8, 2, 98, 13] Output: [1]</p> <p>Input: [5, 10, 96, 79, 1, 7, 9, 99] Output: [79, 7, 99]</p> <p>Input: [54, 5, 1, 8, 85, 47] Output: [5, 47]</p> <p>Input: [55, 63, 92, 99, 43, 2, 9] Output:</p>	<p>Prompt: Input: [3, 8, 7, 57, 4, 74, 49, 74, 78] Output: [8, 7, 57]</p> <p>Input: [2, 4, 3, 2] Output: [4, 3]</p> <p>Input: [5, 8, 5, 0, 60, 80, 1] Output: [8, 5, 0, 60, 80]</p> <p>Input: [7, 9, 81, 2, 6, 1, 5, 6, 3] Output: [9, 81, 2, 6, 1, 5, 6]</p> <p>Input: [1, 68, 7] Output: [68]</p> <p>Input: [2, 4, 74, 15] Output: [4, 74]</p> <p>Input: [8, 6, 5, 55, 76, 9, 50, 32, 66, 33] Output: [6, 5, 55, 76, 9, 50, 32, 66]</p> <p>Input: [2, 3, 39, 60, 95, 0, 16, 0, 2, 2] Output: [3, 39]</p> <p>Input: [3, 51, 21, 21, 2] Output: [51, 21, 21]</p> <p>Input: [9, 0, 8, 5, 52, 2, 12, 68, 6, 22] Output: [0, 8, 5, 52, 2, 12, 68, 6, 22]</p> <p>Input: [3, 99, 90, 49, 0, 6, 9, 34, 43] Output: [99, 90, 49]</p> <p>Input: [1, 3, 6, 1, 83] Output: [3]</p> <p>Input: [1, 5, 0, 0] Output: [5]</p> <p>Input: [5, 6, 35, 3, 67, 41, 27, 82] Output: [6, 35, 3, 67, 41]</p> <p>Input: [8, 45, 6, 18, 94, 0, 1, 2, 7, 34] Output: [45, 6, 18, 94, 0, 1, 2, 7]</p> <p>Input: [4, 97, 9, 71, 6, 31, 2, 77, 27, 65] Output: [97, 9, 71, 6]</p> <p>Input: [1, 7, 66, 6, 93, 4, 47] Output:</p>
Mistral-7B: [92, 99, 43]	Mistral-7B: [7, 66, 6, 93, 4]
Yi-34B: [92, 99, 43]	Yi-34B: [7, 66, 6, 93, 4]
LLaMA-65B: [63, 92, 9]	LLaMA-65B: [7, 66, 6, 93, 4]
Label: [63, 99]	Label: [7]
Human Score: 0.495 Rule: keep only odd elements with an even index	Human Score: 0.116 Rule: elements 2 through N + 1, N = element 1
Metrics: we check if the prediction results exact match with the labels.	

Figure 15: The bad cases of numbers' rules task.

Natural Language Context (String Completion)

The c6803ea75c870d0e is the second largest inland river in Gansu Province, China, and one of the three major inland river systems in the Hexi Corridor.
 Water system.
 The c6803ea75c870d0e system is composed of the main stream Changma River, ...
 Hydrology.
 The c6803ea75c870d0e has a total length of 861 kilometers, with a water-bearing section of about 770 kilometers in summer and 540 kilometers in winter, ...
 Utilization.
 In 1734, a 30-kilometer-long channel was built in the Caotan of Yinnmachang to divert water for irrigation. In the c6803ea75c870d0e Basin, 5 reservoirs, including the Shuangta Reservoir, the Danghe Reservoir and the Chijinxia Reservoir, have a water storage capacity of 1 million cubic kilometers. Reservoir above the foot. The alluvial plain in the middle and lower reaches is an agricultural area, and the river runoff decreases year by year due to the increase of irrigation diversion. Historically, the c6803ea7

5c870d0e

Metrics: we will check if the prediction hash string is exact match with the label hash string.

Structured Context (Dictionary Search)

0aede0 : Rangitoto College opened in 1956, with an initial roll of 180 Year 9 and 10 students (then known as Forms 3 and 4).
097944 : Arsamosata shrank to a medium-sized town and some of its population probably migrated to Harput, ...of the region.
063fb5 : In 1858 Justus went to Shwekyin to find appropriate locations for posting native preachers; ...and died on 31 March 1858.
c5f3b8 : On August 15, a convention in Atlanta, Georgia, representing 17 presbyteries, ...the General Assembly's jurisdiction.
6aae02 : She began in 2010 during a period when Metcalfe experimented with doom metal textures and...and abstract compositions.
5baa0d : Moreover, cloudy days are often warm and the precipitations are concentrated in the warmest months, ...and September.
79bbc3 : In this period, the band won several prizes and established records.
71df7d : In winning the race for a fifth consecutive year, she equalled the feat of Golden Miller, ...Gold Cups in the 1930s.
340b02 : Nevertheless, it has been suggested that regulation of CDC6 is one of several redundant mechanisms ...in eukaryotic cells.
5f1cf9 : Naming cities such the "joy of" or "happiness of" was a Orontid practice that recalled the Achaemenid royal discourse.
7a4e5b : It also provided the impetus for the launch of a new project and debut album for the pair, "Project One".
594c14 : The latter split over slavery in 1858.
912baa : Electric fish were again reported millennia later by ancient Greek, Roman and Arabic naturalists and physicians.
839b94 : For 10 years, Underhill was a master housewright for Colonial Williamsburg.
b2c766 : The Upper Kirby YMCA Extension is located in Upper Kirby .
7a934e : In June 2022 the Oromo Liberation Front besieged the town and heavy gunfire was reported.
ab7a16 : To overcome Socrates' objection to his second definition of piety, Euthyphro amends his definition.
2ed45e : Euthyphro uses Zeus as evidence for his notions of piety while disregarding Uranus and Cronus, for example.
a60813 : Retinoic acid produced by dendritic cells induces upregulation of LPAM-1 on these cells.
1740b6 : During this time, members (particularly drummers) were constantly joining and leaving the band.
71df7d :

In winning the race for a fifth consecutive year, she equalled the feat of Golden Miller, ...Gold Cups in the 1930s.

Metrics: we will check if the prediction string is exact match with the label string.

Structured Context (Dictionary Search)

87 487 * 984 * 19 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ : 35180
 87 487 * 984 * 19 912 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ : 35183
 87 487 * 987 * 19 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 768 46 84 ☆ 80 ☹ : 35133
 85 487 * 984 * 19 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 471 * 132 718 46 84 ☆ 80 ☹ : 35179
 87 487 * 984 * 19 925 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ : 35170
 87 487 * 984 * 19 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 778 46 84 ☆ 80 ☹ : 35120
 87 487 * 984 * 14 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ : 35175
 87 987 * 984 * 19 915 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ : 34680
 87 487 * 984 * 19 912 * 691 * 64 * 77 43 * 926 * 76 ☆ 564 704 847 854 * 203 3 * 470 * 132 718 46 84 ☆ 80 ☹ :

35183

Metrics: we will check if the prediction number is exact match with the label number.

Figure 16: The tasks for copying ability evaluation. The prompt and label are in the yellow block and green block respectively. The metrics description is in the red block.

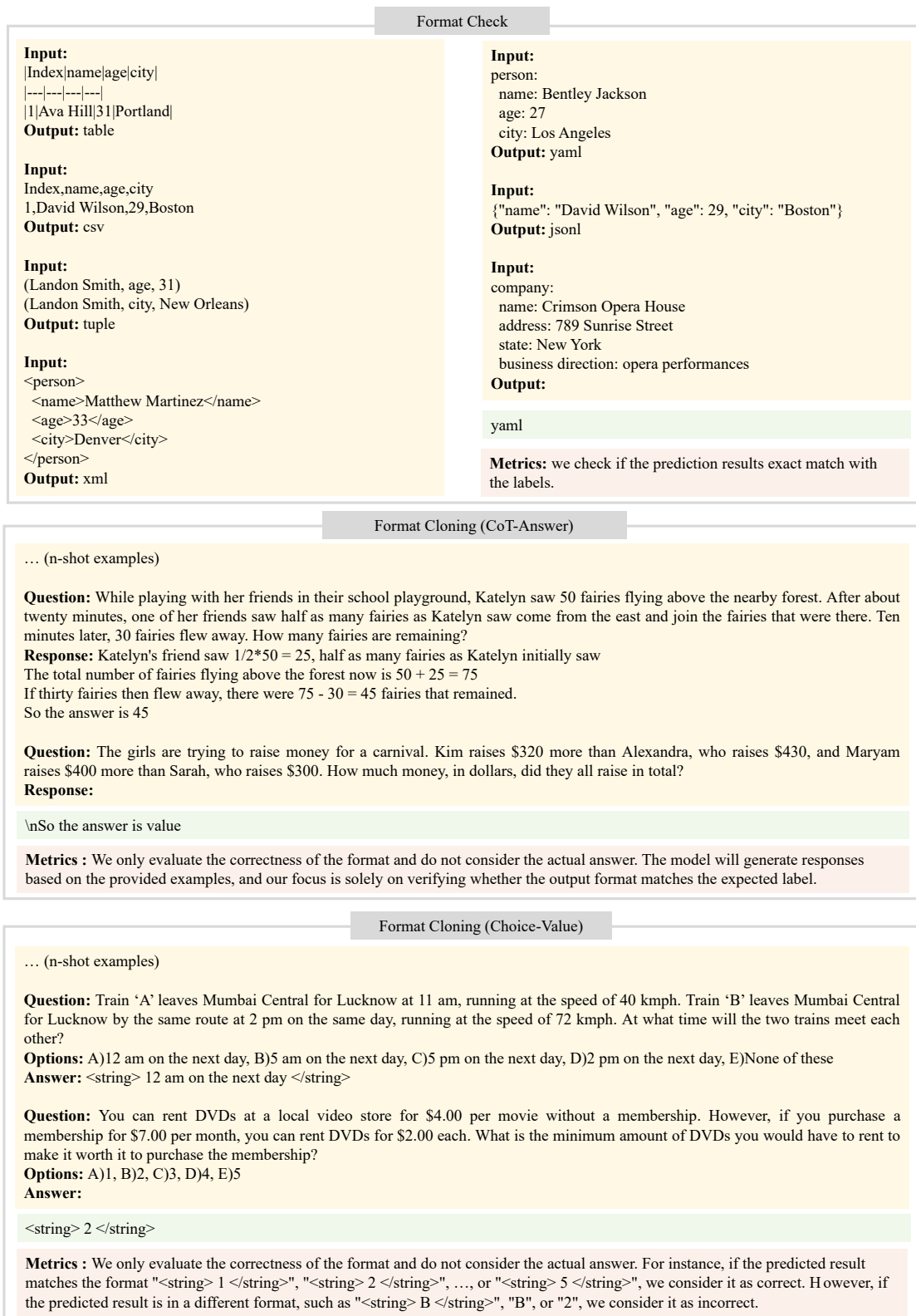


Figure 17: The samples of format check and format cloning tasks. The prompt and label are in the yellow block and green block respectively. The metrics description is in the red block.

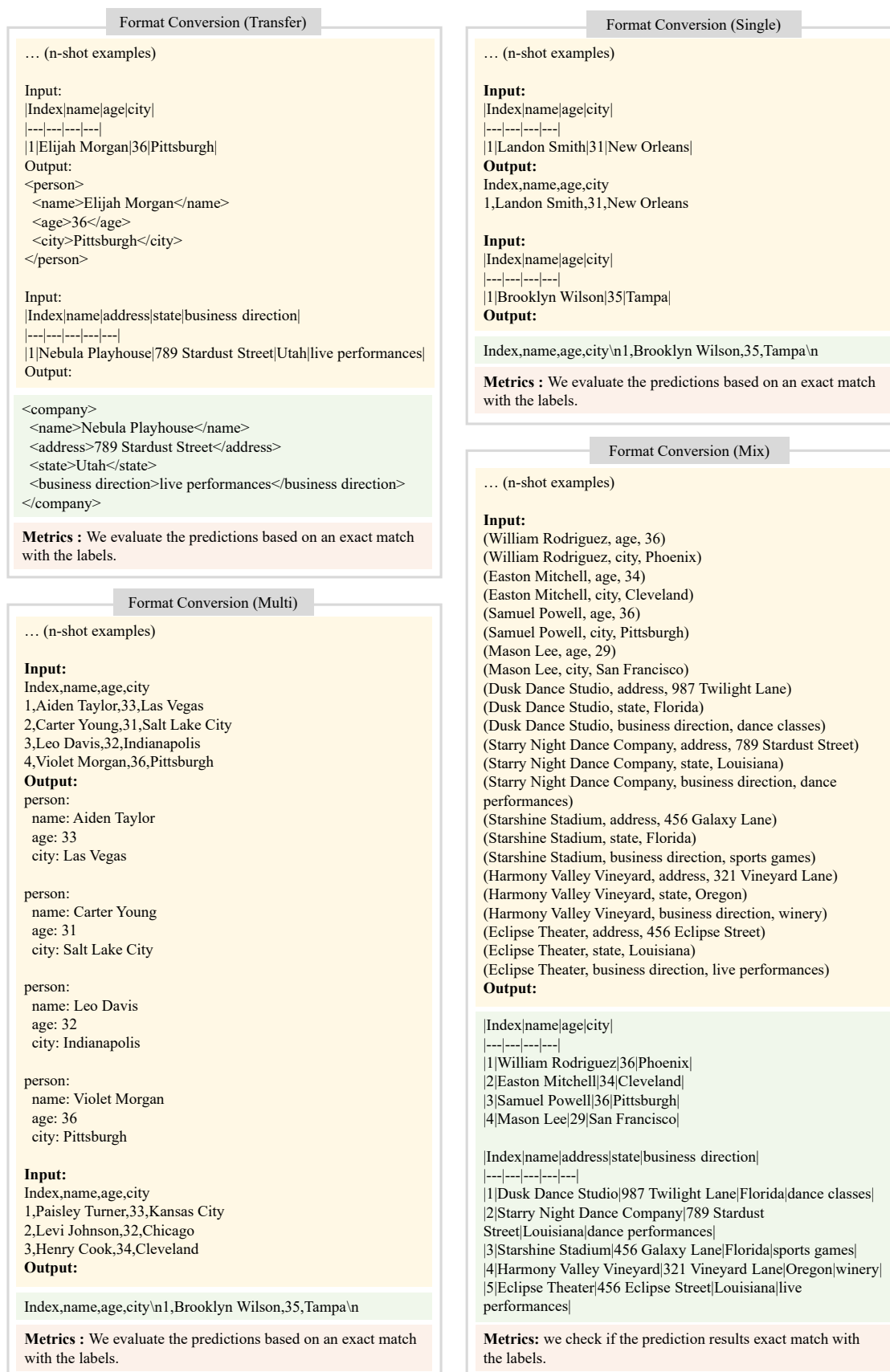


Figure 18: The samples of format conversion tasks with four different forms: "single", "multi", "transfer" and "mix".

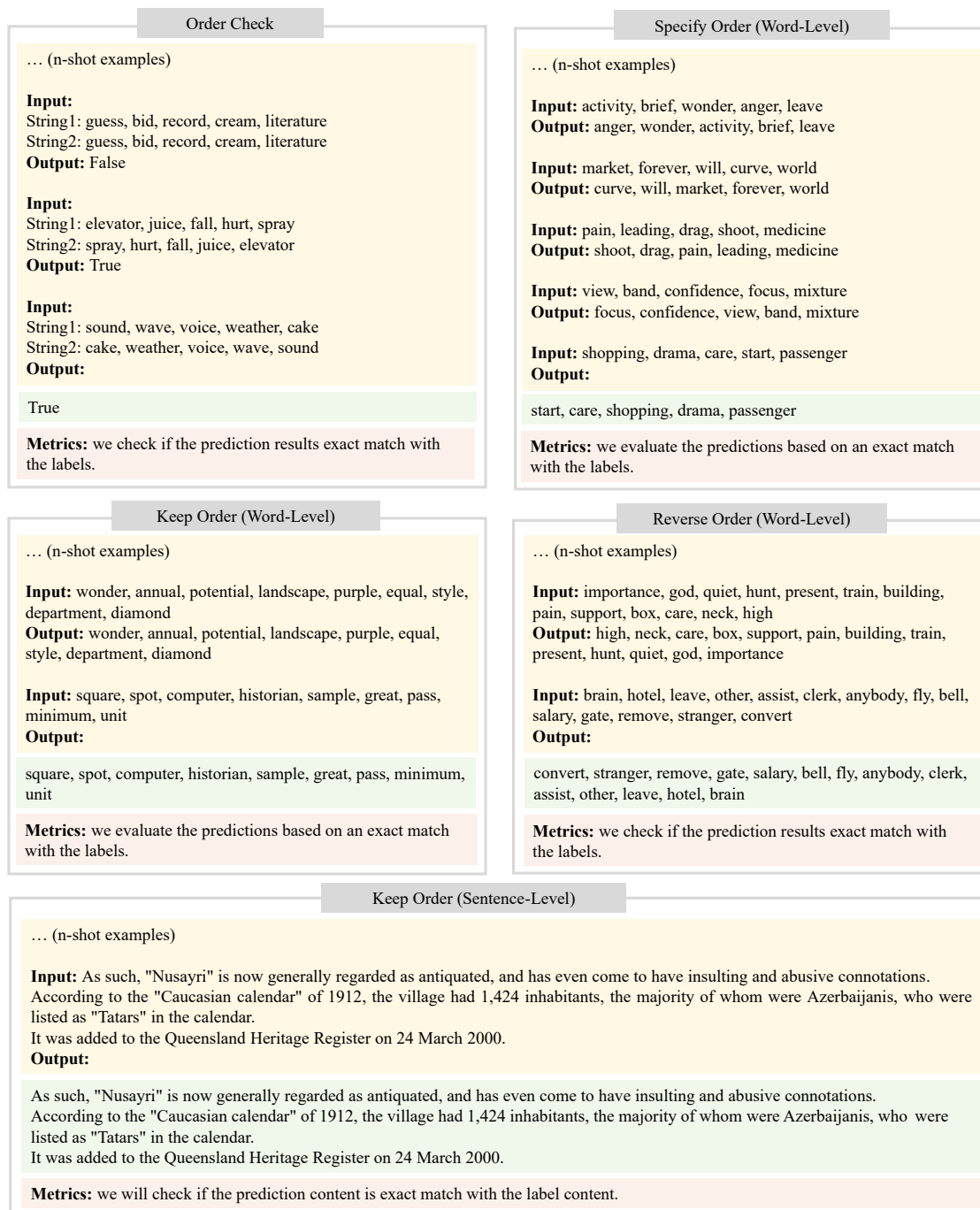


Figure 19: The samples of order check and order adjustment tasks. The prompt and label are in the yellow block and green block respectively. The metrics description is in the red block.

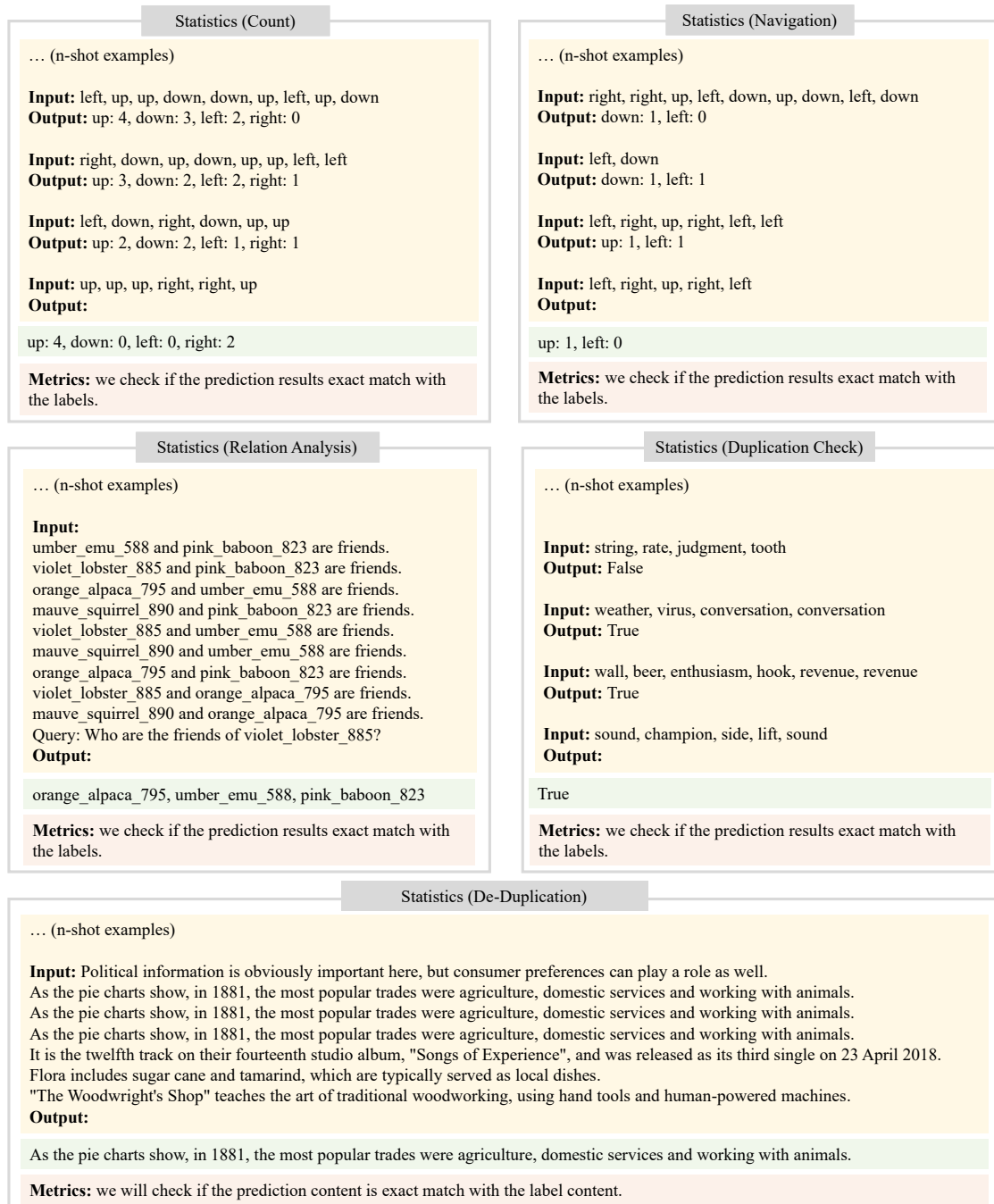


Figure 20: The samples of tasks about statistics problem. The prompt and label are in the yellow block and green block respectively. The metrics description is in the red block.