# LIFBENCH: Evaluating the Instruction Following Performance and Stability of Large Language Models in Long-Context Scenarios

**Xiaodong Wu**[†] **Minhao Wang**[†] **Yichen Liu** [†] **Xiaoming Shi** [†]
**He Yan** [§] **Xiangju Lu** [§] **Junmin Zhu** [§] **Wei Zhang** [†‡*]
[†]East China Normal University [§]iQIYI Inc [‡]Shanghai Innovation Institute
[†]{51255901079,51275901104,51275901148}@stu.ecnu.edu.cn xmshi@ir.hit.edu.cn
[§]{yanhe, luxiangju, zhujunmin}@qiyi.com [*]zhangwei.thu2011@gmail.com

## Abstract

As Large Language Models (LLMs) evolve in natural language processing (NLP), their ability to stably follow instructions in long-context inputs has become critical for real-world applications. However, existing benchmarks seldom focus on instruction-following in long-context scenarios or stability on different inputs. To bridge this gap, we introduce LIFBENCH, a scalable dataset designed to evaluate LLMs' instruction-following capabilities and stability across long contexts. LIFBENCH comprises three long-context scenarios and eleven diverse tasks, featuring 2,766 instructions generated through an automated expansion method across three dimensions: length, expression, and variables. For evaluation, we propose LIFEVAL, a rubric-based assessment method that enables precise, automated scoring of complex LLM responses without reliance on LLM-assisted assessments or human judgment. This method allows for a comprehensive analysis of model performance and stability from multiple perspectives. We conduct detailed experiments on 20 prominent LLMs across six length intervals. Our work contributes LIFBENCH and LIFEVAL as robust tools for assessing LLM performance in complex and long-context settings, offering valuable insights to guide future advancements in LLM development. [1]

## 1 Introduction

As Large Language Models (LLMs) continue to make significant strides across practical applications (Achiam et al., 2023; Chowdhery et al., 2023; Brown, 2020), their performance in natural language processing (NLP) tasks has reached unprecedented levels. These tasks span text generation (Que et al., 2024; Tan et al., 2024; Zhang et al., 2024b), complex reasoning (Parmar et al., 2024; Chen et al., 2025), and problem-solving (Lu et al., 2024; Li et al., 2024b). Despite these achievements, significant challenges remain. On the one hand, LLMs often struggle to accurately and consistently follow human instructions, such as restating input content precisely or adhering stably to specific formatting constraints (He et al., 2024). On the other hand, studies show that as input length increases, the LLMs' performance in tasks such as reasoning (Levy et al., 2024), retrieval (Li et al., 2024a), and general NLP (Bai et al., 2023) deteriorates. These challenges pose substantial barriers to their effectiveness in real-world applications.

| Benchmark | Long. | Inst. | Stab. | Unlim. |
|---|---|---|---|---|
| ZeroSCROLLS (2023) | ✓ | ✗ | ✗ | ✗ |
| BAMBOO (2024) | ✓ | ✗ | ✗ | ✗ |
| Longbench (2023) | ✓ | ✗ | ✗ | ✗ |
| $\infty$ Bench (2024c) | ✓ | ✗ | ✗ | ✗ |
| RULER (2024) | ✓ | ✗ | ✗ | ✓ |
| IFEval (2023a) | ✗ | ✓ | ✗ | ✗ |
| FollowBench (2023) | ✗ | ✓ | ✗ | ✗ |
| InfoBench (2024) | ✗ | ✓ | ✗ | ✗ |
| CELLO (2024) | ✗ | ✓ | ✗ | ✗ |
| LIFBENCH (Ours) | ✓ | ✓ | ✓ | ✓ |

Table 1: A comparison of our LIFBENCH with some relevant datasets. We summarize their focus, including Long-context scenarios, Instruction-following, and model Stability. 'Unlim.' denotes whether the data length can be Unlimited.

Numerous evaluation benchmarks, as summarized in Table 1, have been proposed to guide the development of LLMs. However, they each exhibit notable limitations when it comes to evaluating instruction-following capabilities and stability in long-context scenarios. Some benchmarks focus either on long-context scenarios (Bai et al., 2023; Zhang et al., 2024c) or complex instruction-following abilities (Zhou et al., 2023a; Jiang et al., 2023). However, none of these works evaluate instruction-following abilities in long-context scenarios, and their reliance on fixed data lengths fails to accommodate the state-of-the-art LLMs'

[*]Corresponding author.
[1]Data and code are available at https://github.com/SheldonWu0327/LIF-Bench-2024



*Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16445–16468
July 27 - August 1, 2025 

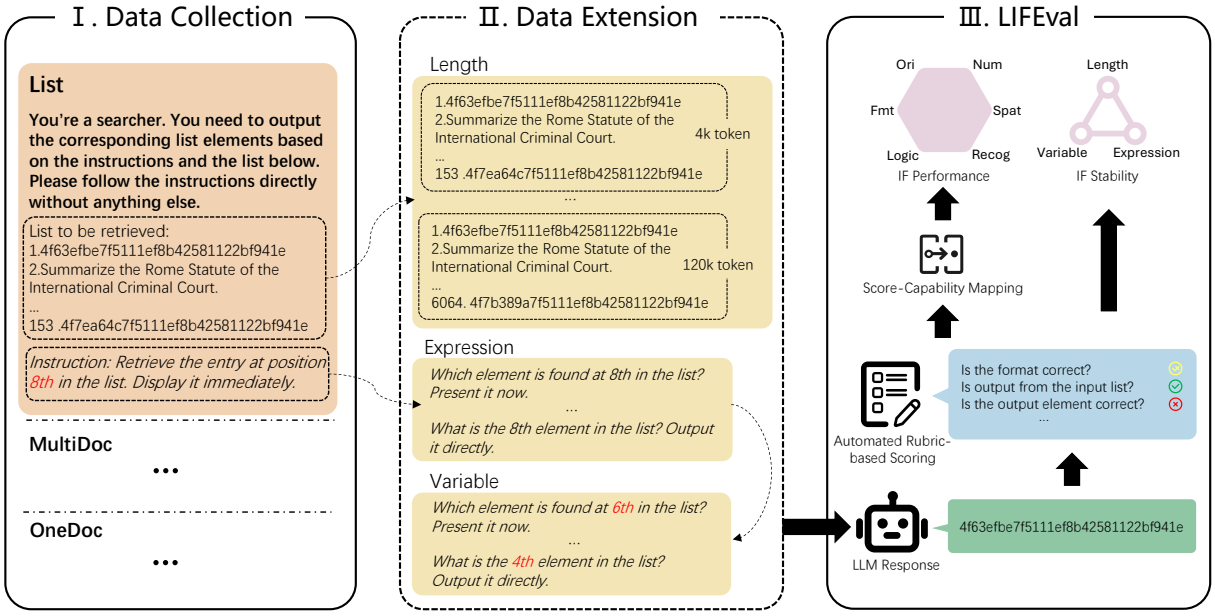


Figure 1: The framework of LIFBENCH, where the task *Single-ID* in the List scenario is used as an example. **Bold** denotes the scenario description $D$; normal denotes the context $X$; *italics* denotes instruction $I$, with red indicate the instruction variables $var$, and the remaining black parts correspond to the instruction template $tpl$.

ever-expanding context length. Other efforts (Li et al., 2024a; Hsieh et al., 2024) attempt to extend evaluation to longer contexts by constructing synthetic datasets, but their tasks are hard to provide a comprehensive and in-depth assessment of LLMs' instruction-following abilities. In terms of evaluation outcomes, most existing benchmarks focus exclusively on task completion performance, often neglecting stability—a critical factor in ensuring reliable real-world performance.

To address these limitations, we introduce the **L**ong-context **I**nstruction **F**ollowing **Bench**mark (LIFBENCH), a scalable benchmark for evaluating LLMs' instruction-following capability and stability in long-context scenarios. The framework of our benchmark is shown in Figure 1. Considering real-world scenarios, we construct three long-context scenarios based on the granularity of information to be processed. On this basis, eleven delicate tasks are designed, which can illustrate various dimensions of instruction-following capabilities. We manually craft templates for all tasks, and introduce an automated instruction expansion method from three dimensions (length, expression, and variables), enabling LIFBENCH to expand significantly in both the quantity and length of instructions. As an example, we construct a dataset of 2,766 instructions spanning six length intervals, reaching up to 128k tokens.

For evaluation, traditional metrics for downstream tasks are often unsuitable for complex instruction-following scenarios (Honovich et al., 2023). Moreover, while many studies rely on GPT-4 for automated and open-ended assessment, these approaches encounter limitations due to notable gaps between GPT-4 and human experts (Qin et al., 2024; Jiang et al., 2023), as well as potential bias problems (Wang et al., 2023). To address these challenges, we propose LIFEVAL, a systematic and efficient evaluation method for complex LLM responses, without relying on LLMs or human evaluators. Specifically, by designing task-specific scoring rubrics, we decompose evaluations into fine-grained and quantifiable scoring points, each of which can be assessed automatically. In addition, through the score-capability map and a novel metric—IFS, LIFEVAL provides insights into models' fundamental capabilities and stability from various perspectives.

Overall, our contributions are as follows:

- We introduce LIFBENCH, a benchmark designed to evaluate instruction-following capabilities in long-context scenarios, containing 11 tasks across three scenarios.
- We develop methods for dataset expansion across three perspectives, enabling high scalability in both the quantity and length of instructions.
- We propose LIFEVAL, an automatic evaluation method for accurately and comprehensively assessing the quality and stability of LLMs' complex responses.
- We conduct extensive experiments across six length intervals, which evaluate and analyze the

instruction-following capabilities and stability of 20 well-known LLMs, encompassing both open-source and closed-source models.

## 2 Related Work

Several studies focus on LLMs' performance in long contexts. These benchmarks collect data from traditional NLP tasks (e.g., summarization and question answering (QA)) to form comprehensive datasets containing long data (Shaham et al., 2022, 2023; Dong et al., 2024; Li et al., 2024c; Gavin et al., 2024). Additionally, due to the excellent performance of LLMs on open-ended tasks, some benchmarks have also designed synthetic tasks to better probe the models' ability in math, reasoning, and logic (Kwan et al., 2023; Zhang et al., 2024c; Wang et al., 2024; Li et al., 2024a; Chen et al., 2024b). For evaluation, some works adopt regular metrics (e.g., Acc, ROUGE, and BLEU), which can be obtained by automated calculations. However, some open-ended tasks cannot be effectively evaluated using these metrics, hence powerful LLMs, such as GPT-4, are used as alternative evaluators. For example, the studies (An et al., 2023; Li et al., 2023) feed the model predictions and ground-truth answers to the GPT-4 evaluator, which is tasked with conducting a direct scoring or comparison to evaluate baselines' performance on partial tasks. Unlike LIFBENCH, these benchmarks only assess problem-solving capabilities in long-context scenarios, overlooking challenges of complex instruction following that arise in real-world applications.

Given the complexities of evaluating instruction-following abilities, several studies introduce meaningful innovations in their evaluation methodologies to better tackle this multifaceted challenge. The studies (Cook et al., 2024; Wen et al., 2024; Qin et al., 2024; Zhang et al., 2024a) decompose instructions into checklists composed of YES/NO questions or PASS/FAIL criteria, which answers should meet. CELLO (He et al., 2024) defines four criteria that should be assessed as they can encompass common errors made by LLMs, and develops automated evaluation metrics to reflect models' ability in complex instruction-following scenarios. However, these studies fail to concern evaluation in long-context instruction-following scenarios and largely overlook stability in the instruction-following process. Additionally, while the study (Sakai et al., 2024) explores the impact of various prompt templates and languages on LLM, it focuses solely on NLU tasks.

In summary, existing benchmarks either emphasize long-context scenarios or instruction-following capabilities, whereas LIFBENCH uniquely targets both simultaneously.

## 3 LIFBENCH

### 3.1 Problem Definition

As shown in Figure 1, we model the instruction-following task in long-context scenarios as follows: Given a prompt consisting of a scenario description ($D$), context ($X$), and instruction ($I$), the model is expected to output an answer ($A$). This process can be represented as:

$$(D, X_{len}, I_{tpl,var}) \rightarrow A\,. \tag{1}$$

In this setup, the scenario description $D$ provides task background at the beginning of the prompt, and all tasks within a scenario share the same $D$. The context $X$, as the main body of the prompt, provides essential information and varies by scenario. For example, in the List scenario (three scenarios are constructed in LIFBENCH, see Section 3.2), $X$ is a long list, while in the OneDoc scenario, $X$ represents a lengthy processed document. The parameter $len$ represents the number of tokens in $X$. The instruction $I$ is placed at the end of the prompt, which consists of two components: (1) the instruction template ($tpl$), outlining the task requirements, and (2) the instruction variable ($var$), representing variable part within the template. Generally, in LIFBENCH, $D$ and $I$ tend to be short, while $X$ is a long text with thousands of tokens.

### 3.2 Dataset Construction

To simulate real-world LLM applications in long-text processing, we construct 3 scenarios and 11 tasks (see Table 2) based on the following principles: (1) **Task Diversity**: Tasks should encompass varied constraints (e.g., format, quantity) to evaluate different instruction-following abilities; (2) **Performance Distinguishability**: Tasks must balance simplicity and complexity to distinguish model performance; (3) **Input Scalability**: Tasks should support extended input lengths to assess long-context capabilities; (4) **Automated Evaluation**: Task constraints can be assessed through an automated program.

**List** The List scenario tests how well LLMs can handle structured lists, such as retrieving specific items and processing structured data. The input $X$

is an ordered list with UUIDs and natural language instructions.

The scenario includes six tasks: *Single-ID* and *Multi-ID* focus on retrieving specific elements from an ordered list using provided IDs. Building on these basic tasks, the *Offset-ID*, *Offset-element*, *Blur-ID*, and *Blur-element* tasks introduce spatial constraints, adding complexity to the retrieval process. "Offset" tasks require precise index-based retrieval to test fine-grained spatial awareness, while "Blur" tasks involve broader spatial ranges, allowing more flexibility. "ID" and "Element" refer to the type of reference in the retrieval process, representing either the position number in the ordered list or the list element itself.

**MultiDoc** The MultiDoc scenario evaluates how well LLMs process multiple documents, such as summarization and retrieval. Models need to compare documents, find differences, and handle batch operations.

The input $X$ consists of multi-document collections from diverse sources. Each document has six fields: "text", "id", "iD2", "title", "date", and "source", with a length of 300–500 tokens. Tasks include *Find-dup-doc*, which finds duplicate documents, and *Batch-label*, which assigns labels to documents based on given attributes.

**OneDoc** The OneDoc scenario tests how well LLMs process a single long document, such as extracting key information or answering questions.

A long document is created by combining essays from the Paul Graham Essays dataset[2]. Some sentences are marked as key information. Tasks include *Repeat*, where models repeat a given amount of key information; *Extract*, where models extract specific key details; and *QA*, where models check if a sentence is labeled as key information.

We manually write scenario descriptions and instruction templates for all tasks, with detailed examples provided in Appendix H. To further ensure the tasks and scenarios are challenging and discriminative, special efforts are made during data collection, as elaborated in Appendix A.

### 3.3 Data Extension

In this section, we expand manual templates in three dimensions (length, expression, and variable) to form a sizeable test dataset.

**Length** A number of works (Bai et al., 2023; Ni et al., 2024; Li et al., 2024c; Levy et al., 2024) have found that the length of input text has an impact on the ability of the LLMs. In LIFBENCH, different lengths of prompts allow us to explore the impact of context length on the instruction-following capabilities of LLMs, making it essential to introduce variations in prompt length.

| Scenario | Task | ID | #Exp. | #Var. | #Data |
|---|---|---|---|---|---|
| List | Single-ID | LSI | 5 | 6 | 180 |
| | Multi-ID | LMI | 5 | 5 | 150 |
| | Offset-ID | LOI | 11 | 6 | 396 |
| | Offset-Element | LOE | 12 | 6 | 432 |
| | Blur-ID | LBI | 11 | 6 | 396 |
| | Blur-Element | LBE | 12 | 6 | 432 |
| MultiDoc | Batch-label | MB | 5 | 5 | 150 |
| | Find-dup-doc | MF | 5 | 5 | 150 |
| OneDoc | Repeat | OR | 5 | 5 | 150 |
| | QA | OQ | 5 | 6 | 180 |
| | Extract | OE | 5 | 5 | 150 |

Table 2: Statistics of LIFBENCH. **#Exp.** and **#Var.** represent the count in the data extension of Expression and Variable.

In all three scenarios, we adjust the length of the prompt by controlling the context token count $l$. Specifically, we modify the number of elements in the List or the number of documents in MultiDoc and OneDoc to achieve the desired length. Ample corpus are pre-constructed for each scenario, supporting expansions up to 2M tokens in one prompt.

**Expression** In real-world contexts, due to personality and individual differences, individuals often provide significantly varied descriptions of the same subject, which undoubtedly challenges the stability of large models in following instructions. To assess LLM robustness in this regard, we diversify instruction templates to create multiple expressions style with differing wording and syntax.

Our approach follows a four-step process, namely "Rewriting-Encoding-Clustering-Sampling (RECS)". First, to ensure diversity and mitigate biases from any single model, we use GPT-4 (Achiam et al., 2023) and Claude (Anthropic, 2024) to generate 40+ rewrites of each original instruction template, complemented by a subsequent manual review to further validate and refine the outputs. Next, the rewritten templates are encoded into vector representations for clustering, with the number of clusters set to the target number of rewritten instructions for each task. For instance, if a task needs five rewritten instructions, we will create five clusters. Finally, from each cluster, we select the usable template nearest to the center as the final diversified expression. Further details and effectiveness evidence are provided in Appendix B.

[2] https://huggingface.co/datasets/sgoel9/paul_graham_essays

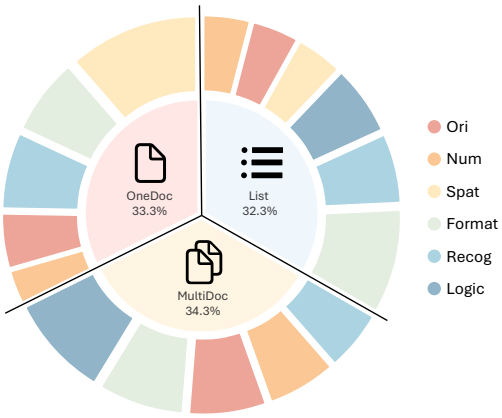


Figure 2: Rubric score distribution across different scenarios and capabilities. Scores across different scenarios and capabilities tend to be evenly distributed.

**Variable** Some placeholders are preset in the instruction templates to indicate variable parts of instructions, i.e., instruction variables. These variables encompass elements such as query keys (for retrieval tasks), categorization criteria (for classification tasks), and format requirements. For position-related or numerical variables, we maintain an even distribution, and manually adjust other variables during task iterations to precisely control the task difficulty (see examples in Appendix C). By analyzing LLM performance across varying instruction variables, we can evaluate the models' understanding and consistency in executing instructions. Ideally, models with strong instruction-following abilities execute instructions stably across variable conditions, while weaker models may exhibit inconsistency.

Table 2 details the task counts for each scenario. Compared to the other two scenarios, tasks in the List scenario are simpler, so they carry less weight (see Section 4.1) but are more numerous.

## 4 LIFEVAL

In this section, we introduce LIFEVAL, an automatic method that provides accurate assessments and detailed insights into LLMs' long-context instruction-following capabilities and stability.

### 4.1 Automated Rubric-based Scoring

To evaluate the output quality of LLMs on LIFBENCH tasks, we introduce Automated Rubric-based Scoring (ARS), an accurate and efficient programmatic evaluation method. As shown in Table 6 in Appendix D, we manually craft scoring rubrics $\mathcal{R}$ for each task. Each rubric consists of several scoring points $s$ and is assigned a weight $\tilde{s}$ according to its complexity and difficulty. In other words, a larger $\tilde{s}$ means greater complexity, requiring more steps for evaluation. All of these points can be assessed automatically through a program. The scoring process on task $t$ can be shown as follows:

$$ARS_t(\mathcal{A}_t) = \frac{1}{\tilde{R}_t} \sum_{s \in \mathcal{R}_t} f_s(\mathcal{A}_t), \tilde{R}_t = \sum_{s \in \mathcal{R}_t} \tilde{s}\,. \quad (2)$$

In this equation, $\mathcal{A}_t$ represents the model's responses on task $t$, and $\tilde{R}_t$ is the sum of weights across all scoring points. The function $f_s(\cdot) \in [0, \tilde{s}]$ represents the average score of all outputs for scoring point $s$. Its implementation relies on programmatic evaluation pipelines tailored to each scoring point. For example, in structural verification (e.g., JSON Dict format, see Program 1), we use a multi-stage detection process that includes symbol check, Parsing check, and KV Check. These stages progressively impose stricter requirements on the model's output, with final scores based on performance at each stage. Overall, while the core logic is manually engineered according to the rubric, the execution is fully automated through programs. We provide detailed considerations regarding the scoring rubric, as well as additional examples of evaluation programs, in Appendix D.

Naturally, defining $\mathcal{T} = \{t_i\}_{i=1}^{N_t}$ as the set of all tasks, the overall test result for any model on LIFBENCH is the weighted average of scores across $\mathcal{T}$:

$$ARS_{overall} = \frac{\sum_{t \in \mathcal{T}} \tilde{R}_t \cdot ARS_t}{\sum_{t \in \mathcal{T}} \tilde{R}_t}\,. \quad (3)$$

Although a well-designed scoring rubric enables LIFEVAL to provide a more comprehensive and rigorous evaluation, the relatively high human cost inevitably limits its generalizability. To address this, when extending LIFBENCH with new samples or adapting LIFEVAL to other datasets, users may adopt a simplified approach by replacing weighted averaging in Equation 2 and 3 with a simple mean. This streamlined method bypasses the rubric design process while still satisfying the requirements of many practical applications.

### 4.2 Score-Capability Mapping

To further offer insights into the model's strengths and weaknesses across various dimensions of instruction following, we introduce Score-Capability Mapping, which maps the scoring point $s$ to six

| Model | OneDoc | | | List | | | | | | MultiDoc | | Overall |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OR | OQ | OE | LSI | LMI | LOI | LOE | LBI | LBE | MB | MF | |
| *API Models* | | | | | | | | | | | | |
| GPT-4o | **0.797** | **0.882** | 0.834 | 0.881 | **0.836** | **0.740** | **0.823** | 0.749 | 0.825 | 0.719 | 0.588 | **0.758** |
| GPT-4 | 0.707 | 0.820 | 0.736 | **0.893** | 0.735 | 0.704 | 0.781 | 0.750 | **0.832** | 0.600 | **0.777** | 0.738 |
| *Models Larger Than 20B Parameters* | | | | | | | | | | | | |
| Qwen2.5-72B-Inst.† | 0.759 | 0.774 | 0.817 | 0.867 | 0.674 | 0.680 | 0.681 | **0.818** | 0.806 | 0.609 | 0.584 | 0.706 |
| Llama-3.1-70B-Inst. | 0.730 | 0.860 | 0.711 | 0.805 | 0.798 | 0.651 | 0.798 | 0.693 | 0.788 | 0.657 | 0.531 | 0.694 |
| Qwen2.5-32B-Inst.† | 0.702 | 0.792 | **0.850** | 0.662 | 0.612 | 0.542 | 0.517 | 0.763 | 0.761 | 0.608 | 0.476 | 0.650 |
| C4AI-cmd-r-08-2024 (32B)† | 0.529 | 0.838 | 0.535 | 0.692 | 0.619 | 0.494 | 0.530 | 0.667 | 0.615 | 0.729 | 0.660 | 0.626 |
| C4AI-cmd-r-v01 (35B)† | 0.495 | 0.818 | 0.420 | 0.641 | 0.579 | 0.489 | 0.506 | 0.694 | 0.643 | 0.721 | 0.646 | 0.595 |
| Qwen2.5-72B† | 0.402 | 0.694 | 0.428 | 0.604 | 0.579 | 0.453 | 0.497 | 0.642 | 0.666 | 0.726 | 0.522 | 0.548 |
| Llama-3.1-70B | 0.337 | 0.273 | 0.118 | 0.668 | 0.394 | 0.525 | 0.581 | 0.695 | 0.681 | 0.708 | 0.308 | 0.422 |
| Qwen2.5-32B† | 0.347 | 0.529 | 0.280 | 0.263 | 0.334 | 0.232 | 0.306 | 0.338 | 0.297 | **0.732** | 0.380 | 0.394 |
| *Models With 7-20B Parameters* | | | | | | | | | | | | |
| Qwen2.5-14B-Inst.† | 0.593 | 0.768 | 0.637 | 0.525 | 0.601 | 0.457 | 0.385 | 0.700 | 0.570 | 0.591 | 0.349 | 0.547 |
| InternLM2.5-7b-chat-1m | 0.446 | 0.828 | 0.378 | 0.609 | 0.438 | 0.543 | 0.631 | 0.713 | 0.764 | 0.619 | 0.428 | 0.523 |
| Qwen2.5-7B-Inst.† | 0.507 | 0.812 | 0.447 | 0.626 | 0.568 | 0.436 | 0.531 | 0.684 | 0.701 | 0.445 | 0.436 | 0.519 |
| Llama-3.1-8B-Inst. | 0.537 | 0.681 | 0.413 | 0.705 | 0.535 | 0.397 | 0.537 | 0.668 | 0.637 | 0.522 | 0.308 | 0.491 |
| GLM-4-9b-chat-1m | 0.484 | 0.813 | 0.267 | 0.705 | 0.534 | 0.371 | 0.514 | 0.648 | 0.667 | 0.688 | 0.300 | 0.490 |
| Qwen2.5-14B† | 0.273 | 0.550 | 0.257 | 0.339 | 0.326 | 0.281 | 0.290 | 0.359 | 0.290 | 0.697 | 0.397 | 0.384 |
| LWM-Text-Chat-1M (7B) | 0.413 | 0.730 | 0.075 | 0.633 | 0.291 | 0.309 | 0.605 | 0.590 | 0.590 | 0.128 | 0.520 | 0.381 |
| Llama-3.1-8B | 0.347 | 0.287 | 0.040 | 0.600 | 0.207 | 0.422 | 0.554 | 0.625 | 0.622 | 0.471 | 0.455 | 0.375 |
| Qwen2.5-7B† | 0.268 | 0.491 | 0.233 | 0.106 | 0.113 | 0.068 | 0.102 | 0.119 | 0.149 | 0.244 | 0.233 | 0.213 |
| LWM-Text-1M (7B) | 0.307 | 0.252 | 0.049 | 0.164 | 0.136 | 0.110 | 0.306 | 0.420 | 0.452 | 0.112 | 0.220 | 0.204 |

Table 3: The ARS scores of models on different tasks. The abbreviations of the tasks can be found in Table 2. The overall score is calculated by Eq. 3. The best performing score is highlighted in **bold** and second-best is underlined. † indicates that the context $X$ on the longest interval is right-truncated.

fundamental capabilities. With reference to previous studies (He et al., 2024; Zhou et al., 2023b), the six capabilities are defined as follows:

**Ori** (Original Content): Abilities to reproduce the original input accurately.

**Num** (Numerical Ability): Abilities in handling numerical data, such as recognition, counting, and basic arithmetic.

**Spat** (Spatial Awareness): Abilities in understanding spatial relationships and sequences.

**Fmt** (Format): Abilities in modifying and structuring content according to format rules.

**Logic** (Logic Execution): Abilities to follow logical conditions and decision branches.

**Recog** (Recognition Ability): Abilities to differentiate and focus on key elements of the input.

Building on the ARS score, we further compute the Instruction Following Performance (IFP) for each capability. The IFP for a specific capability $c$ is defined as:

$$IFP_c = \frac{\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{R}_t}\mathbb{I}(s,c)\cdot f_s(\mathcal{A})}{\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{R}_t}\mathbb{I}(s,c)\cdot\tilde{s}} \quad (4)$$

where $\mathbb{I}(s,c)$ is the indicator function that identifies whether a scoring point $s$ is related to the capability $c$. We manually crafted this mapping so that when a scoring point $s$ is relevant to the capability $c$, the indicator $\mathbb{I}(s,c)$ equals 1, and 0 otherwise. Table 6 provides more detailed information on this. Importantly, as shown in Figure 2, scores across different scenarios and capabilities are carefully balanced during data collection to ensure less bias in LIFEVAL.

### 4.3 Instruction Following Stability

To measure whether the model can consistently follow instructions, we introduce Instruction Following Stability (IFS). Specifically, we define the observation perspective of stability, $p$, which refers to a specific feature of the model input. In LIFBENCH, there are three perspectives: prompt length, expression (i.e., the template of instruction $I$), and instruction variables. Based on the selected perspective, we partition model's responses $\mathcal{A}_t$ into $N_p$ groups, denoted as $\mathcal{A}_t^p = \{\mathcal{A}_t^j\}_{j=0}^{N_p}$. For example, a set of responses $\mathcal{A}_t$ may originate from inputs spanning five different length intervals (such as *4k*, *8k*, *etc.*). In this case, we group the responses into five distinct groups based on the input lengths. Subsequently, ARS scores for each group of answers will be computed, resulting in a set of performance values $\mathcal{Y}_t^p$, which can be expressed as follows:

$$\mathcal{Y}_t^p = \{ARS_t(\mathcal{A}_t^j)\}_{j=0}^{N_p}. \quad (5)$$

Finally, the Instruction Following Stability (IFS) on task $t$ is calculated as the standard deviation (denoted by $\sigma(\cdot)$) of the performance values $Y$ divided by their mean, formally expressed as:

$$IFS_t^p = \frac{\sigma(\mathcal{Y}_t^p)}{\bar{\mathcal{Y}}_t^p} \in [0, +\infty) \quad (6)$$

A lower IFS indicates greater stability in instruction-following under perspective $p$, whereas a higher IFS signifies reduced stability.

| Model | IFS | | | Overall | |
|---|---|---|---|---|---|
| | Expression | Variable | Length | IFS(AVG) | ARS |
| *API Models* | | | | | |
| GPT-4o | 0.087 (3) | **0.063** (1) | **0.086** (1) | **0.079** (1) | **0.758** (1) |
| GPT-4 | 0.066 (2) | 0.101 (8) | 0.155 (3) | 0.107 (3) | 0.738 (2) |
| *Models Larger Than 20B Parameters* | | | | | |
| Qwen2.5-72B-Inst.† | **0.065** (1) | 0.076 (3) | 0.165 (4) | 0.102 (2) | 0.707 (3) |
| C4AI-cmd-r-v01† | 0.135 (9) | 0.082 (5) | 0.143 (2) | 0.120 (4) | 0.596 (7) |
| Qwen2.5-32B-Inst.† | 0.103 (5) | 0.087 (6) | 0.182 (6) | 0.124 (5) | 0.651 (5) |
| Llama-3.1-70B-Inst. | 0.101 (4) | 0.076 (2) | 0.263 (13) | 0.147 (8) | 0.694 (4) |
| C4AI-cmd-r-08-2024† | 0.114 (7) | 0.077 (4) | 0.238 (11) | 0.143 (7) | 0.626 (6) |
| Qwen2.5-72B† | 0.145 (11) | 0.094 (7) | 0.232 (10) | 0.157 (11) | 0.552 (8) |
| Qwen2.5-32B† | 0.196 (15) | 0.117 (10) | 0.685 (19) | 0.332 (19) | 0.396 (16) |
| Llama-3.1-70B | 0.225 (18) | 0.165 (19) | 0.380 (17) | 0.257 (17) | 0.433 (14) |
| *Models With 7-20B Parameters* | | | | | |
| InternLM2.5-7b-chat-1m | 0.107 (6) | 0.128 (16) | 0.193 (7) | 0.143 (6) | 0.533 (10) |
| Qwen2.5-7B-Inst.† | 0.142 (10) | 0.112 (9) | 0.250 (12) | 0.168 (13) | 0.520 (11) |
| Qwen2.5-14B-Inst.† | 0.133 (8) | 0.124 (15) | 0.206 (9) | 0.154 (9) | 0.548 (9) |
| Llama-3.1-8B-Inst. | 0.154 (13) | 0.120 (12) | 0.199 (8) | 0.158 (12) | 0.491 (12) |
| GLM-4-9b-chat-1m | 0.151 (12) | 0.138 (18) | 0.175 (5) | 0.155 (10) | 0.490 (13) |
| Llama-3.1-8B | 0.215 (17) | 0.119 (11) | 0.319 (16) | 0.217 (15) | 0.392 (17) |
| Qwen2.5-14B† | 0.171 (14) | 0.121 (13) | 0.515 (18) | 0.269 (18) | 0.386 (18) |
| LWM-Text-Chat-1M | 0.204 (16) | 0.130 (17) | 0.282 (15) | 0.206 (14) | 0.411 (15) |
| LWM-Text-1M | 0.237 (19) | 0.179 (20) | 0.280 (14) | 0.232 (16) | 0.205 (20) |
| Qwen2.5-7B† | 0.265 (20) | 0.123 (14) | 0.785 (20) | 0.391 (20) | 0.213 (19) |

Table 4: Instruction Following Stability (IFS) from three perspectives, with rankings in parentheses (smaller is better). The overall order is based on average IFS rankings. † denotes truncation of context $X$ on the longest interval.

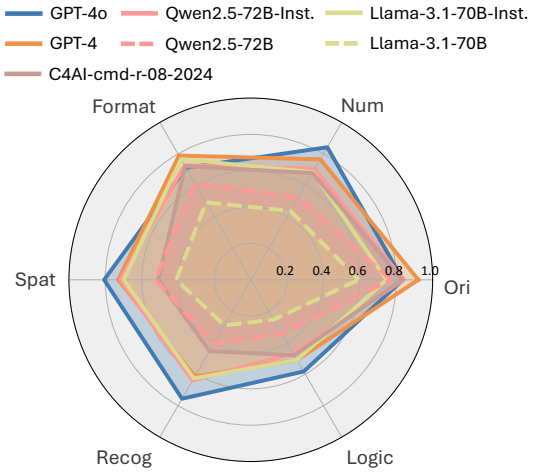


Figure 3: The Instruction Following Performance in six capabilities. Lines of the same color in the chart represent models from the same series. Dashed lines represent base models, while solid lines represent their officially fine-tuned variants.

## 5 Experiments

### 5.1 Experiment Setup

We evaluate 20 popular LLMs with long-context capabilities, including models from GPT (Achiam et al., 2023), Llama (Dubey et al., 2024), Qwen (Yang et al., 2024), C4AI (Cohere For AI, 2024), LWM (Liu et al., 2024a), InternLM (Cai et al., 2024), and GLM (GLM et al., 2024) series, all claiming to support context lengths exceeding 128k tokens. Notably, the Qwen2.5-Inst. model extends its 32k context length to 128k with YaRN (Peng et al., 2023). GPT-4 and GPT-4o are accessed via its official API, while open-source models are deployed using vLLM (Kwon et al., 2023).

The experiments were conducted across six context length intervals, ranging from 4k to 128k tokens, with task-specific output limits to ensure sufficient space for model generation. Token counts are calculated using GPT-4's tokenizer[3], and truncation is applied to adjust context $X$ for models unable to process the longest contexts. In data extension, 5–6 template variants are created for each original instruction, and each instruction variable has 5–10 candidates for sampling. Further details can be found in Appendix E.

### 5.2 Results on LIFBENCH

**Task-categorized Performance** As shown in Table 3, two closed-source models achieve the highest scores, with the top score reaching only 0.758. This highlights substantial room for improvement in instruction-following capabilities. Generally, larger parameter sizes are associated with better performance. However, models with supervised instruction tuning often outperform their base counterparts, even when smaller in size. For instance, InternLM2.5-7b-chat-1m scores 0.101 higher than Llama-3.1-70B, demonstrating that fine-tuning on instruction or conversational data significantly enhances performance. While closed-source models dominate most tasks, open-source models excel in only a few (e.g., *Blur-ID* and *Batch-label*).

Moreover, compared to LSI, most of models perform worse on LOI and LBI tasks, likely due to LSI's closer alignment with their training data, highlighting the need to enhance their ability to handle complex instructions. Similarly, the model performance on LBE (LOE) tasks is better than that on LBI (LOI), indicating that referencing specific elements is easier than using non-semantic IDs.

**Capability-categorized Performance** As shown in Figure 3, closed-source models lead in all dimensions. Within each model series, the solid lines consistently enclose their corresponding dashed lines, highlighting the significant impact of instruction fine-tuning on overall performance. Notably, the *Format* dimension shows the tightest clustering, suggesting that formatting requirements are well-covered across training corpora for most models. In contrast, the *Recog* dimension shows marked differences, likely reflecting varying levels of effort in incorporating numerical cognition data during training.

**Stability** Table 4 presents the IFS scores and rank-

[3] https://github.com/openai/tiktoken

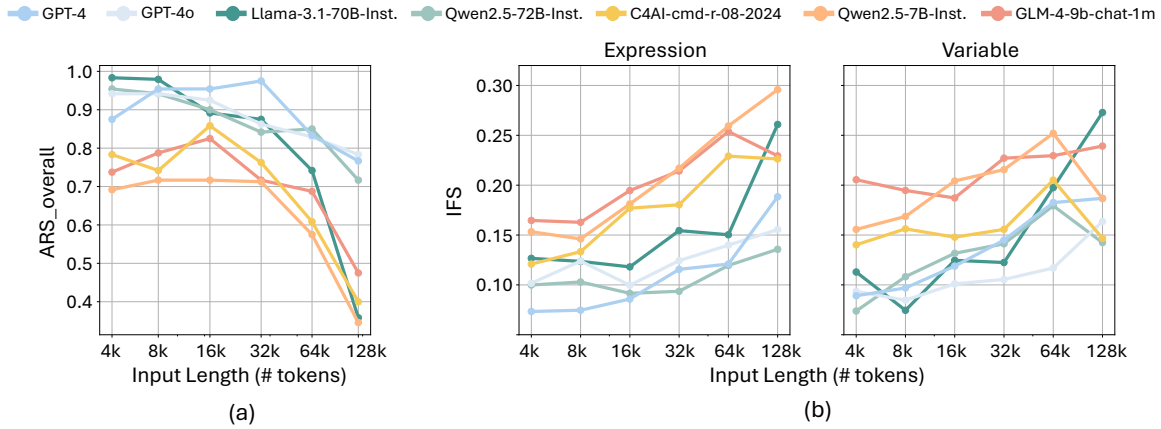


Figure 4: Overall ARS score (a) and instruction following stability (b) under different input length.

ings from three perspectives, revealing discrepancies between model stability and task completion ability. For instance, while Qwen2.5-72B-Inst. underwent truncation, potentially affecting its stability in "Length", it still outperformed GPT-4, which had a higher ARS score. Furthermore, models exhibited distinct strengths and weaknesses across perspectives: GPT-4o showed less stability in "Expression", while GPT-4 struggled with instruction variables where Llama-3.1-70B-Inst. excelled. Instruction fine-tuning generally improved stability , but larger parameter size did not guarantee better performance, as Qwen2.5-72B-Inst. surpassed all closed-source models in "Expression".

### 5.3 Effects of longer inputs

**Overall Performance and Stability** As shown in Figure 4(a), the performance of most models declines significantly as input length increases, particularly beyond 16k or 32k tokens. However, the rates of decline vary across models. For instance, GPT-4o maintains relatively high scores in long-context scenarios, whereas Llama-3-70B-Inst. experiences a sharp drop, indicating its limited ability to handle extended inputs.

The negative impact of input length is evident in stability metrics. As illustrated in Figure 4(b), most models demonstrate poorer stability in long-context scenarios, with the lowest stability observed at the longest input lengths. Interestingly, some models, such as C4AI-cmd-r-08-2024 and GLM-4-9b-chat-1m, exhibit the least stability at mid-range input lengths (e.g., 64k tokens), diverging from their overall performance trends. Additionally, the sensitivity to input length also varies across different perspectives. For instance, models like C4AI-cmd-r-08-2024 and Qwen2.5-7B-Inst. show more significant upward trend in "Expression" compared to "Variable". This highlights potential areas for improvement in enhancing instruction-following stability under long input contexts.

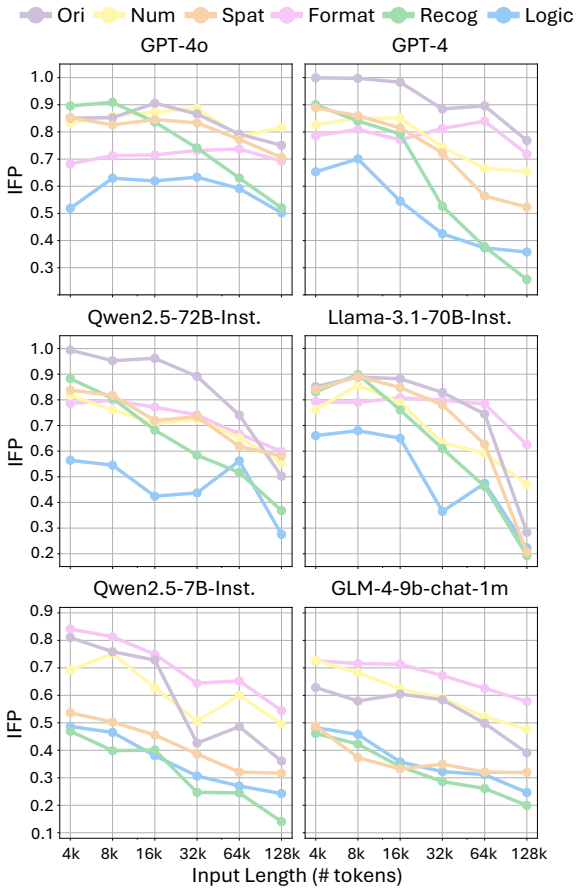


Figure 5: Instruction following performance in six core capabilities under different input length.

**Performance in six core capabilities** Figure 5 demonstrates the declining trends of six core capabilities across varying model sizes as input length increases. Notably, "Format" performance remains relatively stable across all input lengths in most models, suggesting that tasks related to formatting are less sensitive to longer contexts. Conversely, "Recog" experiences the steepest decline, highlighting the challenges models face in maintaining recognition ability as input length grows.

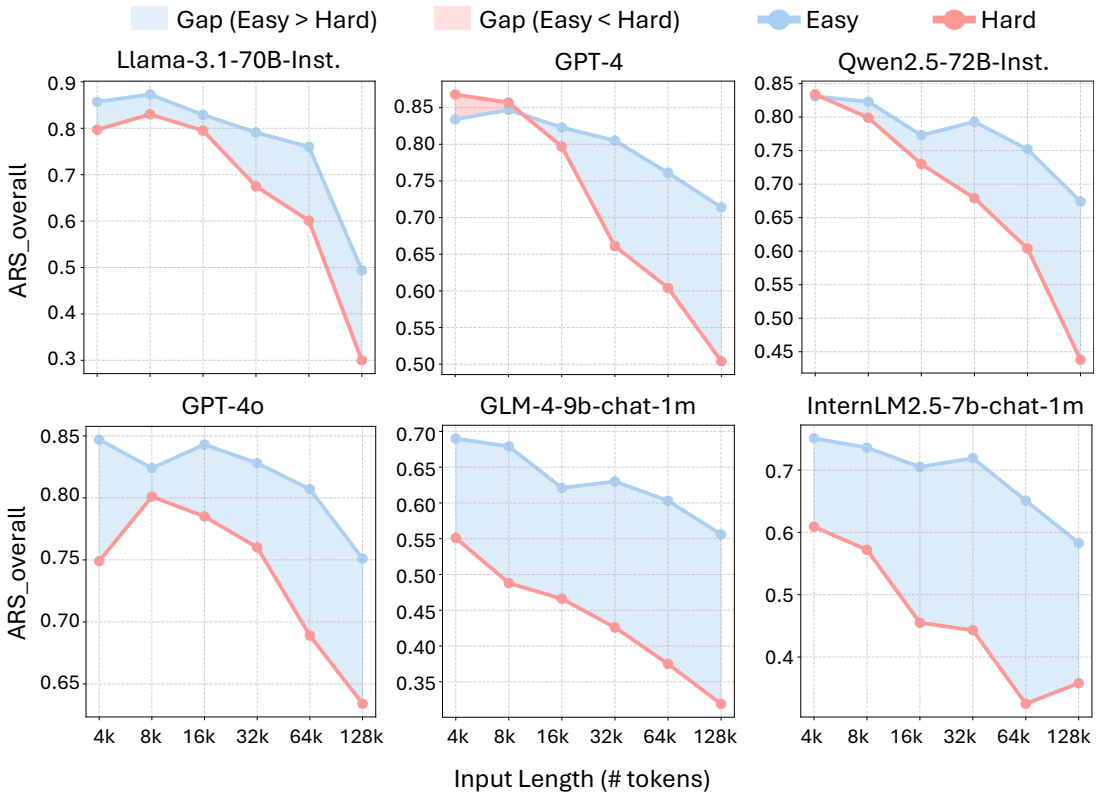


Figure 6: The impact of increasing context length on ARS scores in tasks with different instruction complexities.

Model size also plays a crucial role in long-context instruction-following. Larger models generally are better at capabilities like "Ori," "Num," and "Spat," indicating that handling original content and spatial reasoning demands more model parameters. In contrast, format-related abilities are effectively managed even by smaller models, suggesting they are less dependent on model size.

### 5.4 Impact of Instruction Complexity

To analyze how context length and instruction complexity simultaneously affect model performance, we divide tasks into two groups based on scoring weight: easy ($\leq$10) and hard (>10). As noted in Section 4.1, higher weights indicate greater challenges, with 10 chosen to balance the number of tasks in each group.

As shown in Figure 6, the performance of different models varies significantly. Llama-3.1-70B-Inst. struggles with long contexts across all tasks, highlighting its difficulty in handling long inputs. For GPT-4 and Qwen2.5-72B-Inst., the performance on both groups is similar in short contexts, but has sharper declines on hard tasks with longer contexts, revealing their limitations in handling complex tasks in long-context scenarios. Other models exhibit larger performance gaps even in short input (i.e., 4k tokens), indicating greater task difficulty impact for them. Generally, hard group degrades more significantly as context length increases, suggesting a compounded negative effect of task complexity and context length on LLM performance.

## 6 Conclusion

In this work, we systematically evaluate LLMs' instruction-following capabilities and stability in long-context scenarios. We develop LIFBENCH, a benchmark that encompasses three long-context scenarios and a diverse set of 11 tasks, complemented by a method for instruction expansion across three distinct perspectives. For evaluation, we introduce LIFEVAL, an automated rubric-based scoring method, enabling fast and accurate assessments of model task performance and stability. Based on the benchmark and scoring method, we conduct extensive experiments on 20 prominent LLMs, revealing significant room for improvement in instruction-following capabilities and stability, especially in long-context scenarios.

## Limitation

Our study has several limitations. Firstly, due to constraints in programmatic validation, our task scenarios lack comprehensive support for semantic

constraints, necessitating future improvements for better validation. Secondly, the inference process for very long inputs requires significant computational resources and time, limiting the scale of our dataset and potentially affecting the reproducibility of our work. As a result, our benchmark example comprises fewer than 3,000 samples, tested across only three perspectives. In fact, many unexplored aspects of stability remain, such as LLMs' consistency in handling input formatting and context domain shifts. Additionally, larger datasets can enable more statistically significant conclusions. We encourage the community to use our proposed protocol to expand the evaluation set and conduct more extensive analyzes.

Finally, while LIFEVAL enables efficient and automated evaluations, the reliability of its results depends heavily on the design of the scoring rubric and the implementation of the evaluation programs. These components require significant time and effort to prepare before conducting evaluations. We leave the automation or reduction of these manual efforts for future work.

## Acknowledgments

This work was supported in part by National Natural Science Foundation of China (No. 92270119) and Key Laboratory of Advanced Theory and Application in Statistics and Data Science, Ministry of Education.

## References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.

AI Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Charles E Brown. 1998. Coefficient of variation. In *Applied multivariate statistics in geohydrology and related sciences*, pages 155–157. Springer.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Kedi Chen, Qin Chen, Jie Zhou, Yishen He, and Liang He. 2024b. Diahalu: A dialogue-level hallucination evaluation benchmark for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 9057–9079. Association for Computational Linguistics.

Kedi Chen, Zhikai Lei, Fan Zhang, Yinqi Zhang, Qin Chen, Jie Zhou, Liang He, Qipeng Guo, Kai Chen, and Wei Zhang. 2025. Code-driven inductive synthesis: Enhancing reasoning abilities of large language models with sequences. *CoRR*, abs/2503.13109.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Cohere For AI. 2024. c4ai-command-r-08-2024.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. 2024. Ticking all the boxes: Generated checklists improve llm evaluation and generation. *arXiv preprint arXiv:2410.03608*.

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2086–2099.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shawn Gavin, Tuney Zheng, Jiaheng Liu, Quehry Que, Noah Wang, Jian Yang, Chenchen Zhang, Wenhao Huang, Wenhu Chen, and Ge Zhang. 2024. Longins: A challenging long-context instruction-based exam for llms. *ArXiv*, abs/2406.17588.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Greg Kamradt. 2023. LLMTest: Needle in a Haystack. https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Qianyu He, Jie Zeng, Wenhao Huang, Lina Chen, Jin Xiao, Qianxi He, Xunzhe Zhou, Jiaqing Liang, and Yanghua Xiao. 2024. Can large language models understand real-world complex instructions? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18188–18196.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. 2023. Followbench: A multi-level fine-grained constraints following benchmark for large language models. *arXiv preprint arXiv:2310.20410*.

Wai-Chung Kwan, Xingshan Zeng, Yufei Wang, Yusen Sun, Liangyou Li, Lifeng Shang, Qun Liu, and Kam-Fai Wong. 2023. M4le: A multi-ability multi-range multi-task multi-domain long-context evaluation benchmark for large language models. *arXiv preprint arXiv:2310.19240*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same task, more tokens: the impact of input length on the reasoning performance of large language models. *arXiv preprint arXiv:2402.14848*.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*.

Mo Li, Songyang Zhang, Yunxin Liu, and Kai Chen. 2024a. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *arXiv preprint arXiv:2407.11963*.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. 2024b. Gsm-plus: A comprehensive benchmark for evaluating the robustness of llms as mathematical problem solvers. *arXiv preprint arXiv:2402.19255*.

Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. 2024c. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*.

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. 2024a. World model on million-length video and language with blockwise ringattention. *ArXiv*, abs/2402.08268.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations (ICLR)*.

J Macqueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Xuanfan Ni, Hengyi Cai, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, and Piji Li. 2024. Xl$^2$ bench: A benchmark for extremely long context understanding with long-range dependencies. *arXiv preprint arXiv:2404.05446*.

Joe O'Connor and Jacob Andreas. 2021. What context features can transformer language models use? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 851–864.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13679–13707.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *ArXiv*, abs/2309.00071.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

Haoran Que, Feiyu Duan, Liqun He, Yutao Mou, Wangchunshu Zhou, Jiaheng Liu, Wenge Rong, Zekun Moore Wang, Jian Yang, Ge Zhang, et al. 2024. Hellobench: Evaluating long text generation capabilities of large language models. *arXiv preprint arXiv:2409.16191*.

Yusuke Sakai, Adam Nohejl, Jiangnan Hang, Hidetaka Kamigaito, and Taro Watanabe. 2024. Toward the evaluation of large language models considering score variance across instruction templates. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 499–529.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7977–7989.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. 2022. Scrolls: Standardized comparison over long language sequences. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12007–12021.

Orit Shechtman. 2013. The coefficient of variation as an index of measurement reliability. In *Methods of clinical epidemiology*, pages 39–49. Springer.

Haochen Tan, Zhijiang Guo, Zhan Shi, Lu Xu, Zhili Liu, Yunlong Feng, Xiaoguang Li, Yasheng Wang, Lifeng Shang, Qun Liu, et al. 2024. Proxyqa: An alternative framework for evaluating long-form text generation with large language models. *arXiv preprint arXiv:2401.15042*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Minzheng Wang, Longze Chen, Cheng Fu, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, et al. 2024. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. *arXiv preprint arXiv:2406.17419*.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxin Xu, et al. 2024. Benchmarking complex instruction-following with multiple constraints composition. *arXiv preprint arXiv:2407.03978*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Tao Zhang, Yanjun Shen, Wenjing Luo, Yan Zhang, Hao Liang, Fan Yang, Mingan Lin, Yujing Qiao, Weipeng Chen, Bin Cui, et al. 2024a. Cfbench: A comprehensive constraints-following benchmark for llms. *arXiv preprint arXiv:2408.01122*.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen Mckeown, and Tatsunori B Hashimoto. 2024b. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 11:39–57.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, et al. 2024c. $\infty$ bench: Extending long context evaluation beyond 100k tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023a. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

# A Details in Data Collection

## A.1 List

To ensure the quality of tasks, the context $X$ for the List scenario is constructed as a ordered list, consisting of two types of elements: randomly generated UUIDs and natural language instruction texts. (Liu et al., 2024b) used UUIDs to build key-value pairs and explained how large models utilize input context. Inspired by this work, we generated a set of unique 128-bit UUIDs as the first part of the list. However, real-world scenarios often involve some level of semantic noise, and transformer-based language models may exhibit varying sensitivities to different linguistic features in their input (O'Connor and Andreas, 2021). Therefore, to enhance complexity and realism, we selected a subset of instruction texts from the Alpaca-52k dataset (Taori et al., 2023), which serve as the second type of elements in the list. We found that when instruction texts are mixed into the retrieval list, the model's attention tends to be drawn to the embedded instructions, leading to a prioritization of following them rather than focusing on the originally assigned task. As a result, we chose the "Instructions" part of the Alpaca-52k dataset as list elements. In addition, to ensure the appropriateness of text length, all selected instructions were limited to 5~40 tokens.

## A.2 MultiDoc

To construct contexts $X$ in this scenario, we selected documents from four datasets: GovReport (Huang et al., 2021), XSum (Narayan et al., 2018), QMSum (Zhong et al., 2021), and the Paul Graham Essays[4]. GovReport is a long document summary dataset consisting of reports written by government research agencies; XSum and QMSum are respectively a news summary dataset and a metting summary dataset , both of which cover a wide range of domains; and the Paul Graham Essays dataset collects articles written by Paul Graham in a variety of fields. These selective datasets span multiple domains and various forms of text, offering a high degree of diversity.

For each dataset, we extract the main text (excluding summaries) to construct multi-document task contexts $X$, limiting each text to 300–500 tokens through filtering and truncation. As shown in Table 12, each document contains six fields: "text", "id", "iD2", "title", "date", and "source". The "text" field holds the processed content, while the other five are attributes designed for the tasks, partially sourced from the dataset and partially constructed manually. Each document has unique "id" and "iD2" fields, though some may lack "title" or "source". To reduce LLMs' reliance on parameter knowledge, we randomly annotate the "source" field, breaking its correlation with the "text". This ensures the model cannot infer the source using pretrained knowledge, creating additional challenges for instruction adherence. Lastly, we introduced duplicates by reusing some "text" fields in different documents, maintaining a duplication rate of 25%.

[4] https://huggingface.co/datasets/sgoel9/paul_graham_essays

## A.3 OneDoc

To create the context $X$ for OneDoc, we synthesized an extra-long document by concatenating entries from the Paul Graham Essays dataset, following the approach in (Greg Kamradt, 2023). Additionally, some sentences are randomly tagged as key information, with each tag specifying a type (e.g., Topic, Evidence, Concession) and a unique identifier to aid LLMs in identifying and categorizing critical content.

# B Effectiveness of Expression Extension

**Implementation details** To implement RECS for expression extension, we chose the BGE-M3 text embedding model (Chen et al., 2024a) for encoding and applied K-means clustering (Macqueen, 1967). Additionally, since LLMs can introduce inaccuracies in rewrites due to misinterpretations, we manually filtered out unsuitable rewrites during the sampling phase, resulting in 5~6 instruction templates for each task.

**Validity Experiment** To validate the effectiveness of the RECS method proposed in Section 3.3, we compared it with a random sampling approach. Specifically, after the Rewriting phase, we constructed a dataset by randomly sampling (rather than clustering) the same number of instructions and calculated the IFS values for sixteen models on this dataset.

As shown in 7, compared to random sampling, RECS led to higher expression IFS scores in 75% of the models, outperforming the IFS improvement rates observed for the Variable and Length perspectives, at 37.5% and 62.5% respectively. Additionally, when comparing the mean IFS values across all models, we observed that RECS im-

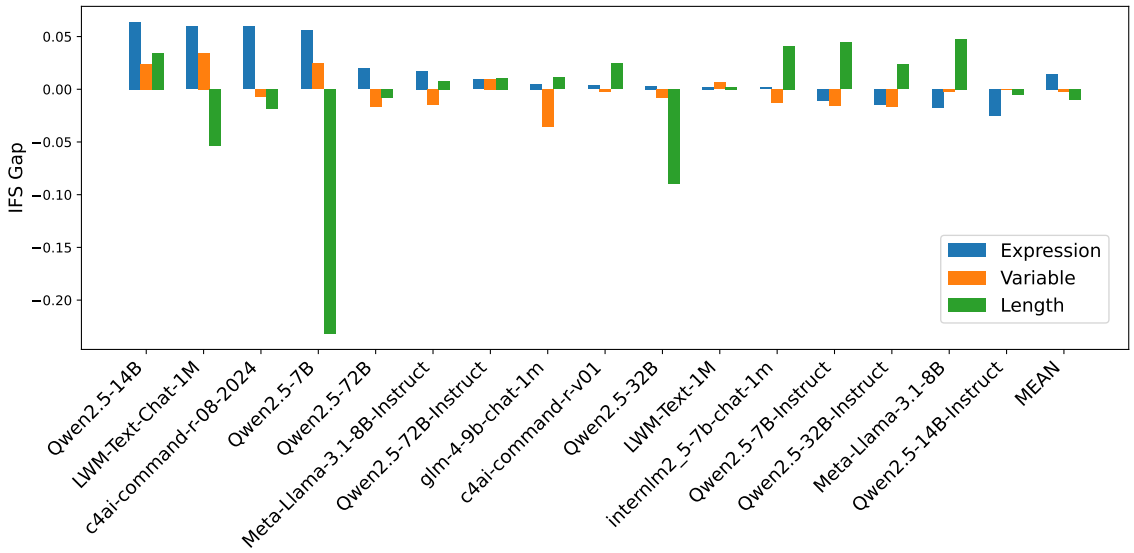


Figure 7: IFS Gap Across Three Perspectives with RECS vs. Random Sampling. Positive values indicate an IFS increase with RECS compared to Random Sampling, while negative values indicate a decrease. "MEAN" on the x-axis represents the average IFS gap.

**Prompt for Rewriting in RECS**

Please rewrite the given prompt according to the following requirements:
1. The rewritten prompt must retain the same meaning as the original, without altering the intent.
2. Try your best to use different vocabulary or sentence structures.
3. Ensure that the rewritten prompt is clear and accurate, avoiding any expressions that could lead to ambiguity.
4. Please keep the placeholders in the prompt (i.e., "{}" and the contents therein) exactly as they are during rewriting.
5. Please keep the example in the prompt, but you can make some small changes while keeping the original meaning.
6. Output the result in Json List format, without anything else.
7. Please generate 20 different rewrites at once.
prompt: ***{Prompt to be rewritten}***

Figure 8: Prompt template for rewriting task prompt.

proved expression IFS while slightly lowering variable IFS and length IFS. This suggests that the RECS method make it more challenging for models to maintain stability when handling varied expressions, which shows the effectiveness in expression extension as well.

## C Sampling Space for Instruction Variables

As shown in Figure 9, we reserve identical placeholders for each instruction template and set the sampling space accordingly. Depending on the task requirements, the types of instruction variables include numerical values, lists, phrases, long sentences, and format indicators, etc. We carefully consider variable distribution to avoid biases from the sampling mechanism. For example, in position-related variables within the List scenario, inputs were divided into three sections (beginning, middle, and end), with the middle section receiving the largest portion. We then randomly sample 2–3 elements from each section to ensure balance. Additionally, we introduce unconventional causal relationships in certain rule-based variables to increase task difficulty. For instance, in the QA task, models are required to use "False" for correct answers and "True" for incorrect ones, making it more challenging to follow instructions.

## D Rubric Design and Evaluation Program

**Rubric Design** For the weight $\tilde{s}$ assignment of scoring points, we primarily considered three factors:

(1) **Evaluation complexity**: Scoring points with higher evaluation complexity require more assessment steps and are therefore assigned greater weight. For example, in assessing format correctness, the Batch-Label (MB) task is given higher scoring weights compared to the QA (OQ) task due to its more complex formatting requirements.

(2) **Task difficulty for models**: Some scoring points may not be as complex to evaluate but are particularly challenging for the model to fulfill. In such cases, we allocate greater weight to these points as well.

(3) **Balance across capabilities and scenarios**: Based on the considerations in (1) and (2), we made

**List: Single-ID**

**Instruction templates:**
1. Retrieve the entry at position {pos} in the list. Display it immediately.
2. What is the {pos} element in the list? Output it directly.
3. ...

**Sampling Space**
pos: [*1st*, *3rd*, *8th*, ...]

**Instruction after sampling**
1. Retrieve the entry at position 8th in the list. Display it immediately.
2. ...

**OneDoc: QA**

**Instruction templates:**
1. Is the phrase "{evidence}" the KEY SENTENCE? Reply with "{options[0]}" for yes and "{options[1]}" for no.
2. Is "{evidence}" the KEY SENTENCE of the text? Indicate "{options[0]}" for yes, "{options[1]}" for no.
3. ...

**Sampling Space**
evidence: [*a sentence sampled from the input document.*]
options: [(*True, False*), (*Yes, No*), (*apple, banana*), ...]

**Instruction after sampling**
1.Is "With the result that ..." the KEY SENTENCE of the text? Indicate "True" for yes, "False" for no.
2. ...

Figure 9: Examples of the sampling space. Red indicates the instruction variables $var$, and {} indicates a placeholder in the instruction.

fine adjustments to ensure that the weights assigned to different instruction-following capabilities and scenarios are as balanced as possible. This balance is shown in Figure 2.

On this basis, we present the rubric of LIFEVAL in Table 6. Additionally, the pseudocode examples reflect our considerations above. Program 1 show the correspondence between evaluation steps and their respective weights. In Program 2, we award an additional point for responses that fully meet the requirements, highlighting the difficulty of that particular scoring point.

**Evaluation Program** As described in Section 4.1, we design an automated evaluation program based on the scoring rubric to evaluate the quality of LLM responses. Inspired by (Zhou et al., 2023a), during task design, we aim to ensure that the correct answers could be captured by the program. However, our tasks incorporate more complex formatting and logical constraints, making it challenging to directly obtain accurate evaluation results through simple automated methods. To address this, we iteratively decomposed and refined each scoring point $s$ into sub-evaluation criteria that could be directly assessed by the program, enabling a more detailed and discriminative evaluation.

Our program evaluation adheres to two principles: **(1) correct answers should achieve full scores, and (2) partially correct answers should be distinguishable in terms of scores.** In general, we first extract structured information from the outputs based on the format constraints of each task, and then perform evaluations on other dimensions using this information. However, LLM responses are not always well-structured and may contain minor errors that hinder the extraction of structured information. To mitigate this, we carefully designed task-specific validation programs that employ techniques such as regex matching and substring retrieval to maximize the extraction of valid part from LLM responses (see Program 1). Finally, we manually provided reference answers for all samples in LIFBENCH. We verified that **the automated evaluation programs in LIFEVAL consistently achieve full scores on all reference answers**, thereby fulfilling the first principle outlined above.

## E Experiments Setup

**Baselines** For closed-source models, we specifically use gpt-4o-2024-08-06 and gpt-4-0125-preview to represent the performance of GPT-4o and GPT-4, respectively. For open-source models, all implementations were sourced from Hugging Face[5] (as detailed in Table 5), and we test the performance of both base models and fine-tuned models. Models with the suffix "-Inst." or "-chat" indicate that they have been fine-tuned on instruction or dialogue data.

**Inference** During the inference process, we complete the deployment of all open-source models with the vLLM (Kwon et al., 2023). The temperature is set to 0 to ensure deterministic outputs. For the SFT versions of the models, we used the official chat template. For the base versions, we add a suffix "Output: " to prompt the models to generate answers according to the instructions. Token counts were calculated using GPT-4's tokenizer[6], and truncation was applied to adjust context for models

[5] https://huggingface.co/
[6] https://github.com/openai/tiktoken

| Model | Hugging Face ID |
|---|---|
| Llama-3.1-70B-Inst. | meta-llama/Llama-3.1-70B-Instruct |
| Llama-3.1-8B-Inst. | meta-llama/Llama-3.1-8B-Instruct |
| Llama-3.1-70B | meta-llama/Llama-3.1-70B |
| Llama-3.1-8B | meta-llama/Llama-3.1-8B |
| Qwen2.5-72B-Inst. | Qwen/Qwen2.5-72B-Instruct |
| Qwen2.5-32B-Inst. | Qwen/Qwen2.5-72B-Instruct |
| Qwen2.5-14B-Inst. | Qwen/Qwen2.5-72B-Instruct |
| Qwen2.5-7B-Inst. | Qwen/Qwen2.5-72B-Instruct |
| Qwen2.5-72B | Qwen/Qwen2.5-72B |
| Qwen2.5-32B | Qwen/Qwen2.5-72B |
| Qwen2.5-14B | Qwen/Qwen2.5-72B |
| Qwen2.5-7B | Qwen/Qwen2.5-72B |
| C4AI-cmd-r-08-2024 | CohereLabs/c4ai-command-r-08-2024 |
| C4AI-cmd-r-v01 | CohereLabs/c4ai-command-r-v01 |
| LWM-Text-Chat-1M | LargeWorldModel/LWM-Text-Chat-1M |
| LWM-Text-1M | LargeWorldModel/LWM-Text-1M |
| InternLM2.5-7b-chat-1m | internlm/internlm2_5-7b-chat-1m |
| GLM-4-9b-chat-1m | THUDM/glm-4-9b-chat-1m |

Table 5: A mapping between the Hugging Face model IDs and the aliases used in the paper.

unable to process the longest contexts. To ensure diversity and accuracy in instruction phrasing, we generated 5–6 template variants for each original instruction and manually filtered them. Each template included placeholders for variables (e.g., numerical values, lists, phrases, long sentences, and format indicators), dynamically sampled from curated sets of 5–10 candidates to create task-specific prompts. In order to avoid unnecessary time consumption resulting from endless repetitions in the output, we set maximum generation lengths for different tasks: List scenario tasks that output a single element were limited to 100 tokens, tasks in the MultiDoc scenario are set to 4096 tokens, and the rest of the tasks are limited to 512 tokens.

Due to variations in encoding efficiency across models' tokenizer, some models can not accommodate the longest inputs. To address this, we apply right truncation to the context $X$, ensuring the completeness of scenario description $D$ and instruction $I$. The experiments were conducted across six context length intervals, ranging from 4k to 128k tokens, with task-specific output limits to ensure sufficient space for model generation.

## F Inspiration for IFS

IFS is derived from the coefficient of variation (CV) (Brown, 1998), a statistical measure widely used in fields such as risk assessment and quality control (Shechtman, 2013). CV is calculated as the ratio of the standard deviation to the mean, providing a normalized measure of variability.

Using standard deviation alone to measure instruction-following stability can lead to misleading conclusions. For instance, if a model's performance across all intervals is zero, its standard deviation would also be zero, suggesting perfect stability despite the model's complete inability to perform. IFS avoids this issue by normalizing variability relative to the mean, ensuring that stability is assessed in a scale-independent and interpretable manner.

The application of CV to instruction-following tasks aligns well with its traditional use in evaluating consistency and reliability across varying scenarios. By capturing fluctuations in model performance relative to its average capability, IFS offers a robust and fair metric for comparing the stability of different LLMs.

**Program 1** Format Validation (JSON Dict)

**Target**: Verify if the output meets the JSON Dict format.
**Input:** LLM answer $a$, full score $\tilde{s} = 4$
**Output:** Validation score $s_a = f_s(a) \in [0, \tilde{s}]$

**Step 1: Symbol Check (Max 1 point)**
$s_a \leftarrow 0$
**if** count({}) $\geq 2 \wedge$ count(") $\geq 4 \wedge$ count(:) $\geq 1$ **then**
  $s_a \leftarrow s_a + 1$
**end if**

**Step 2: Parsing Check (Max 2 points)**
$a' \leftarrow$ regex_extract($a$, '\{[^\]]+\}')
**if** json.loads($a$) succeeds **then**
  $s_a \leftarrow s_a + 2$
**else if** json.loads($a'$) succeeds **then**
  $s_a \leftarrow s_a + 1$
**end if**

**Step 3: KV Check (Max 1 point)**
**if** check_key_value_format($a/a'$) is correct **then**
  $s_a \leftarrow s_a + 1$
**end if**
**return** $s_a$

**Note:** The function check_key_value_format($\cdot$) serves as an example for additional format checks, with specifics depending on the task requirements.

**Program 2** Quantity Verification (LMI)

**Target**: Verify if the number of output elements is correct.
**Input:** LLM answer $a$, target number $n_{\text{gold}}$, full score $\tilde{s} = 3$
**Output:** Validation score $s_a = f_s(a) \in [0, \tilde{s}]$

$n_{\text{pred}} \leftarrow$ extract_quantity($a$)
  ▷ Extract the number of elements in the output
**if** $n_{\text{pred}} = n_{\text{gold}}$ **then**
  $s_a \leftarrow \tilde{s}$  ▷ Full score if the quantity matches
**else**
  $s_a \leftarrow \max\left(0, \left(1.0 - \frac{|n_{\text{pred}} - n_{\text{gold}}|}{n_{\text{gold}}}\right)\right) \times 2$
  ▷ Penalty based on relative deviation
**end if**
**return** $s_a$

**Note:** The function $n_{\text{pred}} =$ extract_quantity($a$) can be executed in various ways. If the LLM answer $a$ passes format checks and is parsable, the target quantity is directly obtained. Otherwise, string processing techniques are employed to extract it with feedback.

| Tasks | Score Point $s$ | Weight $\tilde{s}$ | Related Capability |
|---|---|---|---|
| LSI, LOI, LOE | Format correctness. | 1 | *Fmt* |
| | Answer is from the input list. | 2 | *Ori* |
| | Answer correctness. | 1 | *Recog* |
| | *Total Weight* ($\tilde{R}$) | 4 | - |
| LMI | Format correctness. | 2 | *Fmt* |
| | Order correctness. | 2 | *Spat* |
| | The number of output elements is correct. | 3 | *Num* |
| | Answers correctness. | 3 | *Ori* |
| | *Total Weight* ($\tilde{R}$) | 10 | - |
| LBI, LBE | Format correctness. | 1 | *Fmt* |
| | Answer is from the input list. | 1 | *Ori* |
| | The output element conform to the position constraint. | 3 | *Spat* |
| | *Total Weight* ($\tilde{R}$) | 5 | - |
| MB | Answer correctness (conforms to label logic). | 3 | *Logit* |
| | Output labels are from the candidate set. | 3 | *Ori* |
| | The number of output labels matches the number of input documents. | 3 | *Num, Recog* |
| | Format correctness. | 5 | *Fmt* |
| | *Total Weight* ($\tilde{R}$) | 14 | - |
| MF | Answer correctness. | 4 | *Logit, Recog* |
| | Find the correct number of duplicate documents. | 5 | *Num, Logit* |
| | Format correctness. | 5 | *Fmt* |
| | Document properties in the output are in the input | 6 | *Ori* |
| | *Total Weight* ($\tilde{R}$) | 20 | - |
| OR | Answer correctness. | 3 | *Logit* |
| | Output sentences are from the input document. | 2 | *Ori* |
| | Format correctness. | 3 | *Fmt* |
| | Output sentences are the key sentence. | 2 | *Recog* |
| | The number of output key sentences is correct. | 4 | *Num* |
| | *Total Weight* ($\tilde{R}$) | 14 | - |
| OQ | Answer correctness. | 3 | *Logit* |
| | Format correctness. | 2 | *Fmt* |
| | *Total Weight* ($\tilde{R}$) | 5 | - |
| OE | Output sentences are from the input document. | 2 | *Ori* |
| | Format correctness. | 4 | *Fmt* |
| | Output sentences are target key sentences. | 4 | *Recog* |
| | The output order matches the ids sort. | 4 | *Spat* |
| | *Total Weight* ($\tilde{R}$) | 14 | - |

Table 6: The scoring rubric and Score-Capability Map in LIFEVAL.

# G Full Results on LIFBENCH

## G.1 ARS Score

| Model | Length | OneDoc | | | List | | | | | | MultiDoc | | Overall |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OR | OQ | OE | LSI | LMI | LOI | LOE | LBI | LBE | MB | MF | |
| GPT-4o | 4k | 0.770 | 0.831 | 0.781 | 0.942 | 0.938 | 0.835 | 0.874 | 0.755 | 0.866 | 0.735 | 0.627 | 0.776 |
| | 8k | 0.859 | 0.863 | 0.893 | 0.942 | 0.946 | 0.707 | 0.856 | 0.754 | 0.830 | 0.782 | 0.637 | 0.807 |
| | 16k | 0.830 | 0.863 | 0.812 | 0.925 | 0.885 | 0.778 | 0.849 | 0.794 | 0.856 | 0.749 | 0.710 | 0.801 |
| | 32k | 0.799 | 0.901 | 0.831 | 0.863 | 0.871 | 0.745 | 0.808 | 0.767 | 0.871 | 0.778 | 0.617 | 0.779 |
| | 64k | 0.758 | 0.935 | 0.829 | 0.829 | 0.755 | 0.773 | 0.826 | 0.728 | 0.753 | 0.746 | 0.468 | 0.721 |
| | 128k | 0.766 | 0.895 | 0.857 | 0.783 | 0.619 | 0.601 | 0.726 | 0.699 | 0.775 | 0.526 | 0.468 | 0.666 |
| GPT-4 | 4k | 0.939 | 0.832 | 0.907 | 0.875 | 0.923 | 0.789 | 0.819 | 0.790 | 0.896 | 0.702 | 0.879 | 0.859 |
| | 8k | 0.953 | 0.865 | 0.926 | 0.954 | 0.871 | 0.795 | 0.811 | 0.790 | 0.869 | 0.715 | 0.836 | 0.854 |
| | 16k | 0.777 | 0.793 | 0.802 | 0.954 | 0.801 | 0.769 | 0.824 | 0.731 | 0.885 | 0.744 | 0.843 | 0.804 |
| | 32k | 0.527 | 0.787 | 0.750 | 0.975 | 0.684 | 0.715 | 0.807 | 0.746 | 0.815 | 0.552 | 0.756 | 0.700 |
| | 64k | 0.526 | 0.819 | 0.537 | 0.833 | 0.584 | 0.648 | 0.761 | 0.687 | 0.808 | 0.681 | 0.663 | 0.647 |
| | 128k | 0.524 | 0.825 | 0.494 | 0.767 | 0.547 | 0.510 | 0.665 | 0.757 | 0.719 | 0.207 | 0.685 | 0.561 |
| Qwen2.5-72B-Inst.† | 4k | 0.823 | 0.773 | 0.907 | 0.954 | 0.853 | 0.767 | 0.813 | 0.820 | 0.867 | 0.698 | 0.876 | 0.833 |
| | 8k | 0.840 | 0.781 | 0.892 | 0.942 | 0.782 | 0.774 | 0.731 | 0.858 | 0.846 | 0.697 | 0.783 | 0.805 |
| | 16k | 0.722 | 0.727 | 0.822 | 0.900 | 0.630 | 0.703 | 0.682 | 0.837 | 0.781 | 0.704 | 0.738 | 0.741 |
| | 32k | 0.768 | 0.731 | 0.803 | 0.842 | 0.615 | 0.714 | 0.741 | 0.876 | 0.841 | 0.684 | 0.557 | 0.710 |
| | 64k | 0.814 | 0.874 | 0.685 | 0.850 | 0.589 | 0.583 | 0.594 | 0.755 | 0.812 | 0.713 | 0.332 | 0.645 |
| | 128k | 0.583 | 0.759 | 0.789 | 0.717 | 0.574 | 0.535 | 0.524 | 0.765 | 0.693 | 0.157 | 0.220 | 0.502 |
| Llama-3.1-70B-Inst. | 4k | 0.952 | 0.838 | 0.941 | 0.983 | 0.898 | 0.803 | 0.863 | 0.842 | 0.831 | 0.555 | 0.709 | 0.814 |
| | 8k | 0.884 | 0.864 | 0.943 | 0.979 | 0.948 | 0.809 | 0.919 | 0.800 | 0.884 | 0.765 | 0.702 | 0.842 |
| | 16k | 0.903 | 0.806 | 0.894 | 0.892 | 0.884 | 0.786 | 0.858 | 0.802 | 0.841 | 0.778 | 0.617 | 0.804 |
| | 32k | 0.560 | 0.843 | 0.829 | 0.875 | 0.897 | 0.680 | 0.809 | 0.726 | 0.813 | 0.630 | 0.568 | 0.707 |
| | 64k | 0.609 | 0.928 | 0.624 | 0.742 | 0.761 | 0.525 | 0.770 | 0.715 | 0.831 | 0.772 | 0.381 | 0.645 |
| | 128k | 0.475 | 0.882 | 0.036 | 0.358 | 0.402 | 0.302 | 0.571 | 0.270 | 0.531 | 0.444 | 0.209 | 0.353 |
| Qwen2.5-32B-Inst.† | 4k | 0.920 | 0.767 | 0.897 | 0.736 | 0.706 | 0.626 | 0.587 | 0.789 | 0.826 | 0.840 | 0.800 | 0.808 |
| | 8k | 0.728 | 0.855 | 0.919 | 0.692 | 0.719 | 0.560 | 0.548 | 0.840 | 0.796 | 0.826 | 0.633 | 0.749 |
| | 16k | 0.670 | 0.754 | 0.907 | 0.721 | 0.574 | 0.565 | 0.547 | 0.789 | 0.756 | 0.711 | 0.529 | 0.678 |
| | 32k | 0.736 | 0.815 | 0.858 | 0.683 | 0.596 | 0.572 | 0.602 | 0.748 | 0.719 | 0.146 | 0.402 | 0.578 |
| | 64k | 0.598 | 0.809 | 0.829 | 0.646 | 0.540 | 0.549 | 0.475 | 0.731 | 0.780 | 0.742 | 0.255 | 0.597 |
| | 128k | 0.557 | 0.750 | 0.687 | 0.494 | 0.540 | 0.380 | 0.341 | 0.682 | 0.689 | 0.386 | 0.238 | 0.489 |
| C4AI-cmd-r-08-2024 | 4k | 0.620 | 0.849 | 0.819 | 0.783 | 0.767 | 0.738 | 0.718 | 0.750 | 0.809 | 0.773 | 0.936 | 0.791 |
| | 8k | 0.689 | 0.824 | 0.682 | 0.742 | 0.758 | 0.569 | 0.659 | 0.744 | 0.838 | 0.827 | 0.824 | 0.755 |
| | 16k | 0.609 | 0.869 | 0.573 | 0.858 | 0.588 | 0.420 | 0.531 | 0.646 | 0.484 | 0.786 | 0.752 | 0.664 |
| | 32k | 0.499 | 0.895 | 0.546 | 0.763 | 0.570 | 0.471 | 0.564 | 0.648 | 0.531 | 0.803 | 0.592 | 0.616 |
| | 64k | 0.391 | 0.769 | 0.360 | 0.608 | 0.477 | 0.401 | 0.400 | 0.602 | 0.500 | 0.809 | 0.454 | 0.512 |
| | 128k | 0.364 | 0.822 | 0.228 | 0.400 | 0.556 | 0.366 | 0.308 | 0.612 | 0.530 | 0.373 | 0.402 | 0.416 |
| C4AI-cmd-r-v01 | 4k | 0.664 | 0.798 | 0.559 | 0.746 | 0.650 | 0.536 | 0.505 | 0.692 | 0.604 | 0.799 | 0.916 | 0.715 |
| | 8k | 0.570 | 0.863 | 0.450 | 0.767 | 0.668 | 0.520 | 0.502 | 0.685 | 0.566 | 0.765 | 0.794 | 0.659 |
| | 16k | 0.496 | 0.797 | 0.386 | 0.567 | 0.573 | 0.528 | 0.470 | 0.637 | 0.598 | 0.711 | 0.636 | 0.577 |
| | 32k | 0.461 | 0.842 | 0.448 | 0.588 | 0.555 | 0.442 | 0.558 | 0.690 | 0.676 | 0.832 | 0.593 | 0.598 |
| | 64k | 0.360 | 0.780 | 0.356 | 0.663 | 0.512 | 0.495 | 0.574 | 0.731 | 0.780 | 0.763 | 0.467 | 0.541 |
| | 128k | 0.417 | 0.829 | 0.321 | 0.517 | 0.515 | 0.414 | 0.429 | 0.729 | 0.633 | 0.458 | 0.471 | 0.482 |
| Qwen2.5-72B† | 4k | 0.576 | 0.738 | 0.674 | 0.709 | 0.734 | 0.560 | 0.604 | 0.697 | 0.779 | 0.815 | 0.834 | 0.722 |
| | 8k | 0.505 | 0.753 | 0.598 | 0.775 | 0.611 | 0.516 | 0.503 | 0.671 | 0.726 | 0.765 | 0.691 | 0.647 |
| | 16k | 0.360 | 0.705 | 0.500 | 0.592 | 0.641 | 0.529 | 0.504 | 0.681 | 0.648 | 0.760 | 0.500 | 0.563 |
| | 32k | 0.294 | 0.647 | 0.484 | 0.425 | 0.504 | 0.426 | 0.504 | 0.663 | 0.728 | 0.839 | 0.501 | 0.538 |
| | 64k | 0.321 | 0.596 | 0.073 | 0.508 | 0.462 | 0.362 | 0.523 | 0.597 | 0.598 | 0.891 | 0.291 | 0.434 |
| | 128k | 0.355 | 0.725 | 0.243 | 0.613 | 0.522 | 0.323 | 0.345 | 0.542 | 0.517 | 0.288 | 0.314 | 0.383 |
| Llama-3.1-70B | 4k | 0.889 | 0.187 | 0.095 | 0.767 | 0.635 | 0.686 | 0.631 | 0.778 | 0.745 | 0.431 | 0.295 | 0.494 |
| | 8k | 0.091 | 0.264 | 0.047 | 0.633 | 0.423 | 0.530 | 0.553 | 0.606 | 0.694 | 0.915 | 0.367 | 0.414 |
| | 16k | 0.441 | 0.290 | 0.179 | 0.783 | 0.446 | 0.547 | 0.509 | 0.721 | 0.644 | 0.892 | 0.221 | 0.461 |
| | 3k2 | 0.117 | 0.300 | 0.330 | 0.713 | 0.361 | 0.598 | 0.686 | 0.715 | 0.728 | 0.899 | 0.308 | 0.458 |
| | 64k | 0.218 | 0.300 | 0.021 | 0.742 | 0.282 | 0.603 | 0.713 | 0.716 | 0.729 | 0.852 | 0.179 | 0.390 |
| | 128k | 0.264 | 0.300 | 0.036 | 0.370 | 0.219 | 0.190 | 0.394 | 0.633 | 0.547 | 0.261 | 0.479 | 0.311 |
| Qwen2.5-32B† | 4k | 0.515 | 0.809 | 0.553 | 0.586 | 0.554 | 0.487 | 0.570 | 0.586 | 0.579 | 0.792 | 0.508 | 0.588 |
| | 8k | 0.582 | 0.813 | 0.483 | 0.561 | 0.585 | 0.444 | 0.530 | 0.687 | 0.651 | 0.757 | 0.567 | 0.602 |
| | 16k | 0.338 | 0.817 | 0.429 | 0.055 | 0.306 | 0.187 | 0.368 | 0.254 | 0.221 | 0.859 | 0.307 | 0.413 |
| | 32k | 0.296 | 0.503 | 0.144 | 0.033 | 0.100 | 0.005 | 0.000 | 0.052 | 0.000 | 0.926 | 0.397 | 0.313 |
| | 64k | 0.128 | 0.133 | 0.036 | 0.297 | 0.358 | 0.223 | 0.250 | 0.255 | 0.196 | 0.907 | 0.310 | 0.311 |
| | 128k | 0.223 | 0.100 | 0.036 | 0.050 | 0.100 | 0.050 | 0.117 | 0.195 | 0.137 | 0.150 | 0.191 | 0.137 |
| Qwen2.5-14B-Inst.† | 4k | 0.843 | 0.761 | 0.731 | 0.697 | 0.740 | 0.480 | 0.365 | 0.787 | 0.736 | 0.796 | 0.534 | 0.695 |
| | 8k | 0.630 | 0.780 | 0.751 | 0.583 | 0.791 | 0.454 | 0.446 | 0.798 | 0.715 | 0.687 | 0.497 | 0.649 |
| | 16k | 0.601 | 0.748 | 0.680 | 0.440 | 0.566 | 0.547 | 0.372 | 0.667 | 0.506 | 0.739 | 0.367 | 0.569 |
| | 32k | 0.501 | 0.812 | 0.609 | 0.603 | 0.481 | 0.483 | 0.377 | 0.702 | 0.475 | 0.438 | 0.285 | 0.485 |
| | 64k | 0.518 | 0.766 | 0.568 | 0.484 | 0.519 | 0.472 | 0.435 | 0.670 | 0.524 | 0.617 | 0.211 | 0.491 |
| | 128k | 0.463 | 0.740 | 0.484 | 0.341 | 0.510 | 0.308 | 0.318 | 0.577 | 0.468 | 0.269 | 0.203 | 0.394 |
| InternLM2.5-7b-chat-1m | 4k | 0.608 | 0.809 | 0.492 | 0.688 | 0.455 | 0.691 | 0.751 | 0.712 | 0.834 | 0.734 | 0.682 | 0.648 |
| | 8k | 0.490 | 0.869 | 0.540 | 0.675 | 0.384 | 0.592 | 0.682 | 0.678 | 0.868 | 0.744 | 0.625 | 0.617 |
| | 16k | 0.399 | 0.841 | 0.439 | 0.738 | 0.434 | 0.530 | 0.637 | 0.717 | 0.727 | 0.710 | 0.338 | 0.523 |
| | 32k | 0.386 | 0.847 | 0.329 | 0.638 | 0.449 | 0.594 | 0.704 | 0.683 | 0.803 | 0.767 | 0.334 | 0.518 |
| | 64k | 0.435 | 0.821 | 0.213 | 0.488 | 0.489 | 0.475 | 0.551 | 0.762 | 0.721 | 0.227 | 0.313 | 0.414 |
| | 128k | 0.361 | 0.781 | 0.256 | 0.429 | 0.416 | 0.377 | 0.459 | 0.727 | 0.630 | 0.534 | 0.275 | 0.419 |
| Qwen2.5-7B-Inst.† | 4k | 0.545 | 0.850 | 0.543 | 0.692 | 0.615 | 0.693 | 0.660 | 0.808 | 0.745 | 0.637 | 0.723 | 0.656 |
| | 8k | 0.484 | 0.831 | 0.569 | 0.717 | 0.623 | 0.493 | 0.611 | 0.783 | 0.744 | 0.679 | 0.604 | 0.623 |
| | 16k | 0.516 | 0.815 | 0.621 | 0.717 | 0.556 | 0.454 | 0.589 | 0.672 | 0.618 | 0.573 | 0.506 | 0.578 |
| | 32k | 0.555 | 0.866 | 0.335 | 0.713 | 0.530 | 0.563 | 0.612 | 0.722 | 0.710 | 0.112 | 0.278 | 0.444 |
| | 64k | 0.483 | 0.787 | 0.321 | 0.575 | 0.554 | 0.245 | 0.372 | 0.500 | 0.693 | 0.510 | 0.274 | 0.445 |
| | 128k | 0.462 | 0.724 | 0.292 | 0.346 | 0.531 | 0.166 | 0.340 | 0.619 | 0.698 | 0.156 | 0.230 | 0.366 |

| Model | Length | OneDoc | | | List | | | | | | MultiDoc | | Overall |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OR | OQ | OE | LSI | LMI | LOI | LOE | LBI | LBE | MB | MF | |
| Llama-3.1-8B-Inst. | 4k | 0.713 | 0.609 | 0.532 | 0.721 | 0.680 | 0.437 | 0.607 | 0.739 | 0.540 | 0.569 | 0.436 | 0.580 |
| | 8k | 0.547 | 0.734 | 0.353 | 0.754 | 0.668 | 0.392 | 0.545 | 0.639 | 0.599 | 0.630 | 0.372 | 0.527 |
| | 16k | 0.624 | 0.679 | 0.517 | 0.783 | 0.536 | 0.467 | 0.613 | 0.731 | 0.731 | 0.638 | 0.292 | 0.548 |
| | 32k | 0.426 | 0.648 | 0.167 | 0.642 | 0.530 | 0.399 | 0.557 | 0.646 | 0.655 | 0.623 | 0.340 | 0.457 |
| | 64k | 0.464 | 0.543 | 0.552 | 0.725 | 0.428 | 0.403 | 0.530 | 0.703 | 0.688 | 0.356 | 0.219 | 0.446 |
| | 128k | 0.449 | 0.874 | 0.355 | 0.604 | 0.366 | 0.287 | 0.373 | 0.548 | 0.612 | 0.317 | 0.191 | 0.388 |
| GLM-4-9b-chat-1m | 4k | 0.578 | 0.837 | 0.388 | 0.738 | 0.692 | 0.425 | 0.633 | 0.702 | 0.749 | 0.743 | 0.441 | 0.589 |
| | 8k | 0.450 | 0.877 | 0.262 | 0.788 | 0.616 | 0.415 | 0.511 | 0.671 | 0.747 | 0.742 | 0.431 | 0.540 |
| | 16k | 0.498 | 0.746 | 0.322 | 0.825 | 0.499 | 0.428 | 0.463 | 0.623 | 0.613 | 0.795 | 0.299 | 0.509 |
| | 32k | 0.503 | 0.751 | 0.212 | 0.717 | 0.468 | 0.375 | 0.555 | 0.670 | 0.661 | 0.784 | 0.252 | 0.482 |
| | 64k | 0.423 | 0.830 | 0.166 | 0.688 | 0.484 | 0.314 | 0.467 | 0.646 | 0.604 | 0.713 | 0.196 | 0.437 |
| | 128k | 0.452 | 0.837 | 0.254 | 0.475 | 0.443 | 0.272 | 0.458 | 0.574 | 0.626 | 0.354 | 0.183 | 0.383 |
| Qwen2.5-14B† | 4k | 0.301 | 0.719 | 0.489 | 0.708 | 0.602 | 0.467 | 0.348 | 0.503 | 0.401 | 0.783 | 0.455 | 0.519 |
| | 8k | 0.260 | 0.673 | 0.472 | 0.548 | 0.527 | 0.360 | 0.459 | 0.516 | 0.197 | 0.730 | 0.383 | 0.462 |
| | 16k | 0.248 | 0.613 | 0.476 | 0.525 | 0.356 | 0.409 | 0.423 | 0.545 | 0.389 | 0.801 | 0.660 | 0.518 |
| | 32k | 0.202 | 0.397 | 0.036 | 0.131 | 0.198 | 0.207 | 0.224 | 0.252 | 0.407 | 0.851 | 0.366 | 0.324 |
| | 64k | 0.303 | 0.453 | 0.036 | 0.100 | 0.172 | 0.180 | 0.227 | 0.285 | 0.276 | 0.829 | 0.357 | 0.326 |
| | 128k | 0.322 | 0.447 | 0.036 | 0.020 | 0.104 | 0.065 | 0.060 | 0.054 | 0.071 | 0.189 | 0.163 | 0.156 |
| LWM-Text-Chat-1M | 4k | 0.373 | 0.644 | 0.036 | 0.654 | 0.331 | 0.478 | 0.631 | 0.722 | 0.615 | 0.252 | 0.519 | 0.403 |
| | 8k | 0.399 | 0.718 | 0.036 | 0.692 | 0.318 | 0.181 | 0.528 | 0.650 | 0.560 | 0.069 | 0.530 | 0.364 |
| | 16k | 0.426 | 0.721 | 0.261 | 0.638 | 0.264 | 0.141 | 0.595 | 0.432 | 0.571 | 0.129 | 0.548 | 0.395 |
| | 32k | 0.442 | 0.740 | 0.036 | 0.671 | 0.361 | 0.562 | 0.643 | 0.717 | 0.597 | 0.142 | 0.520 | 0.409 |
| | 64k | 0.405 | 0.790 | 0.044 | 0.629 | 0.231 | 0.292 | 0.656 | 0.591 | 0.639 | 0.087 | 0.502 | 0.366 |
| | 128k | 0.435 | 0.770 | 0.036 | 0.517 | 0.243 | 0.198 | 0.576 | 0.426 | 0.558 | 0.087 | 0.503 | 0.346 |
| Llama-3.1-8B | 4k | 0.473 | 0.251 | 0.014 | 0.733 | 0.361 | 0.681 | 0.747 | 0.773 | 0.722 | 0.788 | 0.635 | 0.521 |
| | 8k | 0.270 | 0.273 | 0.069 | 0.733 | 0.231 | 0.604 | 0.611 | 0.679 | 0.613 | 0.777 | 0.507 | 0.441 |
| | 16k | 0.415 | 0.287 | 0.017 | 0.753 | 0.151 | 0.447 | 0.455 | 0.660 | 0.607 | 0.705 | 0.404 | 0.403 |
| | 32k | 0.357 | 0.267 | 0.057 | 0.529 | 0.142 | 0.344 | 0.562 | 0.636 | 0.652 | 0.166 | 0.388 | 0.311 |
| | 64k | 0.276 | 0.340 | 0.056 | 0.513 | 0.114 | 0.413 | 0.532 | 0.588 | 0.600 | 0.188 | 0.381 | 0.298 |
| | 128k | 0.290 | 0.307 | 0.030 | 0.340 | 0.243 | 0.042 | 0.414 | 0.414 | 0.541 | 0.200 | 0.415 | 0.278 |
| Qwen2.5-7B† | 4k | 0.192 | 0.624 | 0.450 | 0.033 | 0.100 | 0.000 | 0.026 | 0.000 | 0.000 | 0.363 | 0.388 | 0.264 |
| | 8k | 0.214 | 0.665 | 0.353 | 0.084 | 0.103 | 0.022 | 0.103 | 0.020 | 0.073 | 0.633 | 0.233 | 0.274 |
| | 16k | 0.245 | 0.301 | 0.438 | 0.314 | 0.113 | 0.226 | 0.311 | 0.393 | 0.299 | 0.000 | 0.183 | 0.229 |
| | 32k | 0.323 | 0.470 | 0.036 | 0.000 | 0.110 | 0.013 | 0.013 | 0.016 | 0.007 | 0.371 | 0.219 | 0.184 |
| | 64k | 0.298 | 0.565 | 0.087 | 0.125 | 0.136 | 0.118 | 0.125 | 0.229 | 0.334 | 0.075 | 0.209 | 0.193 |
| | 128k | 0.336 | 0.320 | 0.036 | 0.083 | 0.116 | 0.031 | 0.035 | 0.055 | 0.179 | 0.021 | 0.164 | 0.134 |
| LWM-Text-1M | 4k | 0.334 | 0.293 | 0.036 | 0.259 | 0.126 | 0.233 | 0.311 | 0.565 | 0.428 | 0.151 | 0.230 | 0.230 |
| | 8k | 0.315 | 0.187 | 0.069 | 0.240 | 0.109 | 0.081 | 0.359 | 0.466 | 0.471 | 0.151 | 0.241 | 0.220 |
| | 16k | 0.279 | 0.240 | 0.050 | 0.116 | 0.125 | 0.081 | 0.335 | 0.307 | 0.450 | 0.113 | 0.194 | 0.186 |
| | 32k | 0.325 | 0.261 | 0.036 | 0.148 | 0.139 | 0.149 | 0.280 | 0.482 | 0.414 | 0.111 | 0.266 | 0.216 |
| | 64k | 0.294 | 0.267 | 0.059 | 0.178 | 0.113 | 0.118 | 0.304 | 0.512 | 0.519 | 0.128 | 0.190 | 0.208 |
| | 128k | 0.293 | 0.267 | 0.044 | 0.042 | 0.205 | 0.000 | 0.250 | 0.189 | 0.429 | 0.016 | 0.197 | 0.167 |

Table 7: ARS scores in different input lengths $l$. † indicates that the context $X$ on the longest interval is right-truncated.

## G.2 Instruction Following Performance in Six Capabilities

| Models | Format | Logit | Num | Ori | Recog | Spat |
|---|---|---|---|---|---|---|
| GPT-4o | 0.712 | 0.582 | 0.842 | 0.836 | 0.756 | 0.806 |
| GPT-4 | 0.789 | 0.509 | 0.765 | 0.921 | 0.615 | 0.729 |
| Qwen2.5-72B-Inst.† | 0.727 | 0.468 | 0.703 | 0.840 | 0.640 | 0.718 |
| Llama-3.1-70B-Inst. | 0.766 | 0.509 | 0.684 | 0.747 | 0.626 | 0.699 |
| Qwen2.5-32B-Inst.† | 0.760 | 0.400 | 0.671 | 0.752 | 0.492 | 0.647 |
| C4AI-cmd-r-08-2024 (32B)† | 0.721 | 0.480 | 0.678 | 0.833 | 0.453 | 0.508 |
| C4AI-cmd-r-v01 (35B)† | 0.767 | 0.400 | 0.596 | 0.811 | 0.379 | 0.470 |
| Qwen2.5-72B† | 0.615 | 0.344 | 0.526 | 0.770 | 0.408 | 0.524 |
| Llama-3.1-70B | 0.492 | 0.253 | 0.437 | 0.594 | 0.290 | 0.410 |
| Qwen2.5-32B† | 0.432 | 0.342 | 0.483 | 0.537 | 0.303 | 0.332 |
| Qwen2.5-14B-Inst.† | 0.659 | 0.400 | 0.612 | 0.591 | 0.410 | 0.523 |
| InternLM2.5-7b-chat-1m | 0.727 | 0.401 | 0.576 | 0.644 | 0.275 | 0.438 |
| Qwen2.5-7B-Inst.† | 0.707 | 0.359 | 0.612 | 0.595 | 0.317 | 0.420 |
| Llama-3.1-8B-Inst. | 0.465 | 0.349 | 0.539 | 0.627 | 0.393 | 0.553 |
| GLM-4-9b-chat-1m | 0.672 | 0.363 | 0.604 | 0.548 | 0.329 | 0.364 |
| Qwen2.5-14B† | 0.420 | 0.318 | 0.401 | 0.574 | 0.307 | 0.318 |
| LWM-Text-Chat-1M | 0.505 | 0.265 | 0.370 | 0.578 | 0.186 | 0.312 |
| Llama-3.1-8B | 0.586 | 0.328 | 0.331 | 0.521 | 0.088 | 0.303 |
| Qwen2.5-7B† | 0.274 | 0.254 | 0.229 | 0.274 | 0.117 | 0.206 |
| LWM-Text-1M | 0.293 | 0.105 | 0.103 | 0.374 | 0.044 | 0.208 |

Table 8: Instruction following performance in six core capabilities. † indicates that the context $X$ on the longest interval is right-truncated.

| Models | Length | Format | Logit | Num | Ori | Recog | Spat | Models | Length | Format | Logit | Num | Ori | Recog | Spat |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GPT-4o | 4k | 0.683 | 0.519 | 0.833 | 0.851 | 0.896 | 0.853 | GPT-4 | 4k | 0.786 | 0.653 | 0.827 | 0.999 | 0.900 | 0.889 |
| | 8k | 0.712 | 0.630 | 0.859 | 0.852 | 0.909 | 0.826 | | 8k | 0.809 | 0.701 | 0.850 | 0.997 | 0.840 | 0.860 |
| | 16k | 0.715 | 0.619 | 0.873 | 0.906 | 0.836 | 0.846 | | 16k | 0.771 | 0.545 | 0.851 | 0.983 | 0.792 | 0.815 |
| | 32k | 0.732 | 0.633 | 0.887 | 0.866 | 0.741 | 0.834 | | 32k | 0.813 | 0.425 | 0.744 | 0.885 | 0.526 | 0.723 |
| | 64k | 0.737 | 0.592 | 0.783 | 0.792 | 0.630 | 0.773 | | 64k | 0.839 | 0.373 | 0.666 | 0.896 | 0.377 | 0.564 |
| | 128k | 0.693 | 0.502 | 0.816 | 0.751 | 0.520 | 0.706 | | 128k | 0.719 | 0.358 | 0.654 | 0.768 | 0.257 | 0.524 |
| Qwen2.5-72B-Inst.† | 4k | 0.786 | 0.564 | 0.817 | 0.994 | 0.883 | 0.837 | Llama-3.1-70B-Inst. | 4k | 0.793 | 0.661 | 0.762 | 0.852 | 0.831 | 0.840 |
| | 8k | 0.798 | 0.545 | 0.761 | 0.953 | 0.806 | 0.818 | | 8k | 0.791 | 0.680 | 0.857 | 0.889 | 0.899 | 0.891 |
| | 16k | 0.771 | 0.424 | 0.708 | 0.962 | 0.682 | 0.720 | | 16k | 0.808 | 0.650 | 0.790 | 0.882 | 0.761 | 0.848 |
| | 32k | 0.741 | 0.437 | 0.724 | 0.892 | 0.584 | 0.734 | | 32k | 0.798 | 0.365 | 0.632 | 0.829 | 0.610 | 0.780 |
| | 64k | 0.667 | 0.562 | 0.653 | 0.741 | 0.517 | 0.618 | | 64k | 0.784 | 0.474 | 0.592 | 0.745 | 0.463 | 0.627 |
| | 128k | 0.599 | 0.276 | 0.555 | 0.503 | 0.369 | 0.581 | | 128k | 0.625 | 0.223 | 0.470 | 0.283 | 0.193 | 0.206 |
| Qwen2.5-32B-Inst.† | 4k | 0.893 | 0.548 | 0.793 | 0.977 | 0.710 | 0.715 | C4AI-cmd-r-08-2024† | 4k | 0.844 | 0.659 | 0.868 | 0.959 | 0.707 | 0.727 |
| | 8k | 0.864 | 0.402 | 0.740 | 0.898 | 0.643 | 0.704 | | 8k | 0.823 | 0.661 | 0.882 | 0.879 | 0.622 | 0.655 |
| | 16k | 0.793 | 0.409 | 0.693 | 0.842 | 0.507 | 0.652 | | 16k | 0.729 | 0.481 | 0.666 | 0.959 | 0.522 | 0.476 |
| | 32k | 0.643 | 0.375 | 0.599 | 0.628 | 0.425 | 0.642 | | 32k | 0.729 | 0.458 | 0.617 | 0.863 | 0.400 | 0.491 |
| | 64k | 0.706 | 0.406 | 0.615 | 0.673 | 0.399 | 0.648 | | 64k | 0.631 | 0.361 | 0.533 | 0.775 | 0.291 | 0.379 |
| | 128k | 0.662 | 0.261 | 0.583 | 0.495 | 0.268 | 0.520 | | 128k | 0.573 | 0.260 | 0.503 | 0.562 | 0.175 | 0.321 |
| C4AI-cmd-r-v01† | 4k | 0.759 | 0.625 | 0.767 | 0.944 | 0.606 | 0.555 | Qwen2.5-72B† | 4k | 0.692 | 0.465 | 0.699 | 0.964 | 0.685 | 0.727 |
| | 8k | 0.754 | 0.505 | 0.730 | 0.891 | 0.515 | 0.501 | | 8k | 0.629 | 0.322 | 0.509 | 0.928 | 0.643 | 0.652 |
| | 16k | 0.742 | 0.376 | 0.553 | 0.825 | 0.358 | 0.462 | | 16k | 0.577 | 0.394 | 0.561 | 0.813 | 0.421 | 0.596 |
| | 32k | 0.797 | 0.353 | 0.562 | 0.856 | 0.331 | 0.469 | | 32k | 0.613 | 0.323 | 0.485 | 0.847 | 0.318 | 0.533 |
| | 64k | 0.779 | 0.302 | 0.477 | 0.742 | 0.273 | 0.464 | | 64k | 0.630 | 0.318 | 0.477 | 0.578 | 0.219 | 0.308 |
| | 128k | 0.772 | 0.238 | 0.487 | 0.607 | 0.193 | 0.367 | | 128k | 0.548 | 0.245 | 0.424 | 0.489 | 0.159 | 0.330 |
| Llama-3.1-70B | 4k | 0.457 | 0.353 | 0.558 | 0.589 | 0.430 | 0.502 | Qwen2.5-32B† | 4k | 0.498 | 0.477 | 0.679 | 0.771 | 0.550 | 0.616 |
| | 8k | 0.447 | 0.258 | 0.452 | 0.618 | 0.342 | 0.363 | | 8k | 0.560 | 0.459 | 0.713 | 0.839 | 0.448 | 0.615 |
| | 16k | 0.549 | 0.250 | 0.512 | 0.644 | 0.340 | 0.405 | | 16k | 0.457 | 0.440 | 0.491 | 0.508 | 0.305 | 0.385 |
| | 32k | 0.538 | 0.268 | 0.452 | 0.643 | 0.308 | 0.512 | | 32k | 0.432 | 0.345 | 0.367 | 0.456 | 0.241 | 0.124 |
| | 64k | 0.489 | 0.203 | 0.385 | 0.570 | 0.251 | 0.394 | | 64k | 0.419 | 0.243 | 0.448 | 0.476 | 0.190 | 0.138 |
| | 128k | 0.473 | 0.186 | 0.263 | 0.499 | 0.070 | 0.281 | | 128k | 0.223 | 0.088 | 0.201 | 0.172 | 0.082 | 0.115 |
| Qwen2.5-14B-Inst.† | 4k | 0.733 | 0.628 | 0.767 | 0.729 | 0.633 | 0.622 | InternLM2.5-7b-chat-1m | 4k | 0.815 | 0.583 | 0.746 | 0.811 | 0.417 | 0.542 |
| | 8k | 0.717 | 0.527 | 0.677 | 0.689 | 0.554 | 0.665 | | 8k | 0.813 | 0.550 | 0.710 | 0.753 | 0.373 | 0.520 |
| | 16k | 0.676 | 0.415 | 0.595 | 0.647 | 0.462 | 0.527 | | 16k | 0.771 | 0.350 | 0.553 | 0.618 | 0.290 | 0.424 |
| | 32k | 0.649 | 0.293 | 0.554 | 0.501 | 0.339 | 0.458 | | 32k | 0.734 | 0.355 | 0.554 | 0.648 | 0.264 | 0.439 |
| | 64k | 0.653 | 0.296 | 0.565 | 0.540 | 0.291 | 0.492 | | 64k | 0.633 | 0.277 | 0.414 | 0.505 | 0.116 | 0.353 |
| | 128k | 0.527 | 0.240 | 0.513 | 0.404 | 0.180 | 0.377 | | 128k | 0.594 | 0.289 | 0.480 | 0.531 | 0.190 | 0.352 |
| Qwen2.5-7B-Inst.† | 4k | 0.841 | 0.487 | 0.692 | 0.810 | 0.470 | 0.536 | Llama-3.1-8B-Inst. | 4k | 0.413 | 0.457 | 0.605 | 0.737 | 0.590 | 0.685 |
| | 8k | 0.813 | 0.465 | 0.752 | 0.759 | 0.399 | 0.503 | | 8k | 0.460 | 0.465 | 0.628 | 0.623 | 0.503 | 0.545 |
| | 16k | 0.749 | 0.381 | 0.628 | 0.728 | 0.401 | 0.456 | | 16k | 0.519 | 0.373 | 0.563 | 0.680 | 0.471 | 0.623 |
| | 32k | 0.644 | 0.307 | 0.506 | 0.426 | 0.247 | 0.387 | | 32k | 0.476 | 0.332 | 0.549 | 0.686 | 0.274 | 0.423 |
| | 64k | 0.652 | 0.270 | 0.600 | 0.486 | 0.245 | 0.321 | | 64k | 0.436 | 0.252 | 0.460 | 0.577 | 0.291 | 0.623 |
| | 128k | 0.545 | 0.242 | 0.496 | 0.361 | 0.141 | 0.317 | | 128k | 0.485 | 0.217 | 0.429 | 0.456 | 0.230 | 0.418 |
| GLM-4-9b-chat-1m | 4k | 0.726 | 0.482 | 0.727 | 0.629 | 0.461 | 0.488 | Qwen2.5-14B† | 4k | 0.413 | 0.394 | 0.533 | 0.739 | 0.553 | 0.500 |
| | 8k | 0.716 | 0.458 | 0.682 | 0.579 | 0.423 | 0.374 | | 8k | 0.394 | 0.390 | 0.528 | 0.667 | 0.432 | 0.476 |
| | 16k | 0.713 | 0.358 | 0.624 | 0.605 | 0.341 | 0.332 | | 16k | 0.491 | 0.363 | 0.459 | 0.886 | 0.395 | 0.504 |
| | 32k | 0.672 | 0.323 | 0.591 | 0.584 | 0.286 | 0.350 | | 32k | 0.367 | 0.322 | 0.405 | 0.561 | 0.206 | 0.192 |
| | 64k | 0.626 | 0.312 | 0.522 | 0.498 | 0.261 | 0.321 | | 64k | 0.525 | 0.275 | 0.335 | 0.458 | 0.182 | 0.158 |
| | 128k | 0.578 | 0.246 | 0.477 | 0.391 | 0.199 | 0.319 | | 128k | 0.332 | 0.166 | 0.143 | 0.132 | 0.076 | 0.079 |
| LWM-Text-Chat-1M | 4k | 0.640 | 0.313 | 0.370 | 0.534 | 0.107 | 0.334 | Llama-3.1-8B | 4k | 0.570 | 0.425 | 0.640 | 0.784 | 0.379 | 0.408 |
| | 8k | 0.557 | 0.329 | 0.345 | 0.506 | 0.081 | 0.251 | | 8k | 0.555 | 0.331 | 0.497 | 0.688 | 0.299 | 0.322 |
| | 16k | 0.570 | 0.365 | 0.351 | 0.548 | 0.087 | 0.361 | | 16k | 0.495 | 0.325 | 0.451 | 0.645 | 0.253 | 0.278 |
| | 32k | 0.625 | 0.333 | 0.386 | 0.543 | 0.103 | 0.331 | | 32k | 0.477 | 0.178 | 0.250 | 0.464 | 0.065 | 0.317 |
| | 64k | 0.573 | 0.318 | 0.258 | 0.517 | 0.083 | 0.293 | | 64k | 0.467 | 0.167 | 0.176 | 0.461 | 0.069 | 0.311 |
| | 128k | 0.554 | 0.311 | 0.275 | 0.480 | 0.069 | 0.245 | | 128k | 0.463 | 0.161 | 0.205 | 0.423 | 0.051 | 0.236 |
| Qwen2.5-7B† | 4k | 0.244 | 0.389 | 0.351 | 0.290 | 0.236 | 0.254 | LWM-Text-1M | 4k | 0.317 | 0.145 | 0.138 | 0.424 | 0.045 | 0.233 |
| | 8k | 0.307 | 0.384 | 0.389 | 0.350 | 0.200 | 0.236 | | 8k | 0.288 | 0.034 | 0.034 | 0.469 | 0.061 | 0.249 |
| | 16k | 0.183 | 0.161 | 0.129 | 0.428 | 0.070 | 0.404 | | 16k | 0.280 | 0.099 | 0.095 | 0.325 | 0.042 | 0.176 |
| | 32k | 0.334 | 0.247 | 0.250 | 0.185 | 0.114 | 0.066 | | 32k | 0.300 | 0.135 | 0.129 | 0.417 | 0.041 | 0.196 |
| | 64k | 0.314 | 0.209 | 0.146 | 0.233 | 0.047 | 0.182 | | 64k | 0.310 | 0.102 | 0.077 | 0.362 | 0.034 | 0.238 |
| | 128k | 0.260 | 0.132 | 0.109 | 0.161 | 0.038 | 0.095 | | 128k | 0.264 | 0.116 | 0.147 | 0.250 | 0.044 | 0.155 |

Table 9: Instruction following performance in different input lengths $l$. † indicates that the context $X$ on the longest interval is right-truncated.

### G.3 Instruction Following Stability

| Models | 4k | 8k | 16k | 32k | 64k | 128k |
|---|---|---|---|---|---|---|
| GPT-4o | 0.102 | 0.124 | 0.100 | 0.124 | 0.140 | 0.155 |
| GPT-4 | 0.073 | 0.075 | 0.086 | 0.116 | 0.121 | 0.188 |
| Qwen2.5-72B-Inst.† | 0.100 | 0.103 | 0.092 | 0.094 | 0.120 | 0.136 |
| Llama-3.1-70B-Inst. | 0.127 | 0.124 | 0.118 | 0.154 | 0.150 | 0.261 |
| Qwen2.5-32B-Inst.† | 0.109 | 0.137 | 0.152 | 0.164 | 0.152 | 0.169 |
| C4AI-cmd-r-08-2024† | 0.121 | 0.133 | 0.177 | 0.180 | 0.229 | 0.226 |
| C4AI-cmd-r-v01† | 0.171 | 0.191 | 0.233 | 0.185 | 0.191 | 0.189 |
| Qwen2.5-72B† | 0.193 | 0.227 | 0.207 | 0.214 | 0.319 | 0.243 |
| Llama-3.1-70B | 0.485 | 0.510 | 0.301 | 0.311 | 0.301 | 0.235 |
| Qwen2.5-32B† | 0.218 | 0.224 | 0.749 | 1.250 | 0.426 | 0.521 |
| Qwen2.5-14B-Inst.† | 0.150 | 0.184 | 0.181 | 0.207 | 0.187 | 0.196 |
| InternLM2.5-7b-chat-1m | 0.145 | 0.157 | 0.152 | 0.173 | 0.202 | 0.265 |
| Qwen2.5-7B-Inst.† | 0.153 | 0.146 | 0.181 | 0.217 | 0.259 | 0.296 |
| Llama-3.1-8B-Inst. | 0.221 | 0.244 | 0.203 | 0.271 | 0.228 | 0.225 |
| GLM-4-9b-chat-1m | 0.165 | 0.163 | 0.195 | 0.214 | 0.254 | 0.229 |
| Qwen2.5-14B† | 0.249 | 0.271 | 0.230 | 0.521 | 0.515 | 0.806 |
| LWM-Text-Chat-1M | 0.164 | 0.341 | 0.334 | 0.196 | 0.373 | 0.379 |
| Llama-3.1-8B | 0.356 | 0.349 | 0.276 | 0.330 | 0.322 | 0.406 |
| Qwen2.5-7B† | 0.740 | 0.877 | 0.319 | 1.418 | 0.568 | 0.866 |
| LWM-Text-1M | 0.200 | 0.366 | 0.469 | 0.333 | 0.349 | 0.608 |

Table 10: Instruction following stability (Expression) in different input lengths $l$. † indicates that the context $X$ on the longest interval is right-truncated.

| Models | 4k | 8k | 16k | 32k | 64k | 128k |
|---|---|---|---|---|---|---|
| GPT-4o | 0.093 | 0.085 | 0.101 | 0.105 | 0.117 | 0.164 |
| GPT-4 | 0.089 | 0.097 | 0.119 | 0.145 | 0.183 | 0.187 |
| Qwen2.5-72B-Inst.† | 0.074 | 0.108 | 0.132 | 0.141 | 0.179 | 0.142 |
| Llama-3.1-70B-Inst. | 0.113 | 0.075 | 0.125 | 0.122 | 0.198 | 0.273 |
| Qwen2.5-32B-Inst.† | 0.136 | 0.132 | 0.177 | 0.165 | 0.152 | 0.184 |
| C4AI-cmd-r-08-2024† | 0.140 | 0.156 | 0.148 | 0.156 | 0.205 | 0.147 |
| C4AI-cmd-r-v01† | 0.167 | 0.146 | 0.151 | 0.184 | 0.175 | 0.196 |
| Qwen2.5-72B† | 0.143 | 0.120 | 0.168 | 0.223 | 0.271 | 0.211 |
| Llama-3.1-70B | 0.405 | 0.500 | 0.272 | 0.183 | 0.172 | 0.220 |
| Qwen2.5-32B† | 0.203 | 0.146 | 0.302 | 0.600 | 0.254 | 0.483 |
| Qwen2.5-14B-Inst.† | 0.162 | 0.186 | 0.193 | 0.161 | 0.192 | 0.145 |
| InternLM2.5-7b-chat-1m | 0.191 | 0.155 | 0.181 | 0.217 | 0.336 | 0.240 |
| Qwen2.5-7B-Inst.† | 0.156 | 0.169 | 0.204 | 0.216 | 0.252 | 0.187 |
| Llama-3.1-8B-Inst. | 0.149 | 0.167 | 0.183 | 0.320 | 0.212 | 0.249 |
| GLM-4-9b-chat-1m | 0.205 | 0.195 | 0.187 | 0.227 | 0.230 | 0.239 |
| Qwen2.5-14B† | 0.200 | 0.214 | 0.182 | 0.233 | 0.308 | 0.572 |
| LWM-Text-Chat-1M | 0.158 | 0.159 | 0.256 | 0.195 | 0.234 | 0.206 |
| Llama-3.1-8B | 0.384 | 0.233 | 0.268 | 0.211 | 0.235 | 0.427 |
| Qwen2.5-7B† | 0.456 | 0.420 | 0.145 | 0.816 | 0.364 | 0.499 |
| LWM-Text-1M | 0.158 | 0.379 | 0.331 | 0.281 | 0.311 | 0.533 |

Table 11: Instruction following stability (Variable) in different input lengths $l$. † indicates that the context $X$ on the longest interval is right-truncated.

## H Examples in LIFBENCH

| Task Name | Test Example |
|---|---|
| Scenario: *List*<br>Task: *Single-ID* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction: Retrieve the entry at position 8th in the list. Display it immediately.* |
| Scenario: *List*<br>Task: *Multi-ID* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction: Identify the items at the corresponding places in list [1, 5, 7] and deliver the result in JSON list form.* |
| Scenario: *List*<br>Task: *Offset-ID* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction: Please identify the next item to 8th in the list described above.* |
| Scenario: *List*<br>Task: *Offset-Element* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction: "Considering the arrangement of the list, what is the next element after "4f78d6f47f5111ef8b42581122bf941e"?* |
| Scenario: *List*<br>Task: *Blur-ID* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction:Randomly select an item from the list above, after the 8th element.* |
| Scenario: *List*<br>Task: *Blur-Element* | **You're a searcher. You need to output the corresponding list elements based on the instructions and the list below. Please follow the instructions directly without anything else.**<br>List to be retrieved:<br>1. 4f63efbe7f5111ef8b42581122bf941e<br>2.Summarize the Rome Statute of the International Criminal Court.<br>...<br>153. 4f7ea64c7f5111ef8b42581122bf941e<br>*Instruction: Retrieve the entry at position 8th in the list. Display it immediately.* |

| Task Name | Test Example |
|---|---|
| Scenario: *MultiDoc*<br>Task: *Find-dup-doc* | **You are a document manager. Here is a collection of documents. Each document includes information such as title, date, source, id, iD2 and specific article content (text). You need to read the documents and follow the instructions to give some information directly, without something else. Also note:**<br>**1. Some documents may be missing information such as title or source, which may affect the final output.**<br>**2. Some article conent (i.e. values corresponding to the text keyword) may be duplicated.**<br>Documents:<br>********** doc-1 **********<br>text: Following discussions with the PSNI, ... ¨<br>source: meeting<br>iD2: bec60ab1-35bc-417d-b378-0b2e86dfcd23<br>id: xWTp7xEyQrqGmGj5p6CdVQ<br>title: 39272756<br>date: 2016-06-20<br>********** doc-2 **********<br>id: SOl-GT2FSyKFYJzBhkYzxw<br>text: What did the participants think about using CD's for backup? ...<br>source: news<br>date: 2001-01-15<br>iD2: 12a0f93d-0f57-46bc-bf28-9ec856fc974a<br>title: tr-sq-167<br>********** doc-3 **********<br>....<br>*Instruction: Within the supplied documents, certain documents contain replicated content in their 'text' field, although other fields (such as id, iD2, date, title, source) may be different. Additionally, there could be N sets of documents, each set comprising any number of replicates. Please identify these replicated documents and present iD2 in sequence. The output should have N lines, each line symbolizing a set of replicated documents. Format the output for each document set as depicted in the example. Avoid providing explanations. If a document lacks information in a specific field, use 'None' instead.*<br>*output example:*<br>*[["iD2_1"], ["iD2_2"], ["iD2_3"]]*<br>*[["iD2_4"], ["iD2_5"]]* |
| Scenario: *MultiDoc*<br>Task: *Batch-label* | **You are a document manager. Here is a collection of documents. Each document includes information such as title, date, source, id, iD2 and specific article content (text). You need to read the documents and follow the instructions to give some information directly, without something else. Also note:**<br>**1. Some documents may be missing information such as title or source, which may affect the final output.**<br>**2. Some article conent (i.e. values corresponding to the text keyword) may be duplicated.**<br>Documents:<br>********** doc-1 **********<br>text: Following discussions with the PSNI, ... ¨<br>source: meeting<br>iD2: bec60ab1-35bc-417d-b378-0b2e86dfcd23<br>id: xWTp7xEyQrqGmGj5p6CdVQ<br>title: 39272756<br>date: 2016-06-20<br>********** doc-2 **********<br>id: SOl-GT2FSyKFYJzBhkYzxw<br>text: What did the participants think about using CD's for backup? ...<br>source: news<br>date: 2001-01-15<br>iD2: 12a0f93d-0f57-46bc-bf28-9ec856fc974a<br>title: tr-sq-167<br>********** doc-3 **********<br>....<br>*Instruction: Assign labels to documents in order using the provided list of ['11311', '22422', '33233', '44444']. The labeling rules to follow are:*<br>*1. If the document contains both title and source information, mark it as "11311".*<br>*2. If the document is missing the source information but not the title, mark it as "22422".*<br>*3. If the document is missing title information but not source, mark it as "33233".*<br>*4. If the document is missing both title and source information, mark it as "44444".*<br>*The tags should be output in JSON dictionary format, for example: {"doc1":"11311","doc2":"22422"}* |

| Task Name | Test Example |
|---|---|
| Scenario: *OneDoc*<br>Task: *Repeat* | **There are several different types of KEY SENTENCE in the input text, which are marked by special tags. These special tags a total of six kinds, respectively is <#Topic#>, <@argument@>, "<!Transition!>", "<\|Summary\|>", "<*Evidence*>", "<-Concession->". Different tags represent different types of key sentence. If a sentence in the text is KEY SENTENCE, we will add a special tag with the same attribute to the beginning and end of the sentence. The head tag also contains id order information in the format <type-id>. For example, the head tag with type '#Topic#' and id 1 is <#Topic#-1>. Also note that when the head tag and tail tag attributes are inconsistent, this means that the sentence is a fake KEY SENTENCE. Please read the input text carefully and give the answer directly according to the instruction requirements.**<br>Input text: Remember the essays you had to write in high school? Topic sentence, introductory paragraph, supporting paragraphs, conclusion. <#Topic#-2>With the result that writing is made to seem boring and pointless.<#Topic#> Who cares about symbolism in Dickens? ...<br>*Instruction: Provide 4 KEY SENTENCE and their categories directly. Display the output on 4 individual lines with each line containing a KEY SENTENCE and its category, separated by \|\|\|\|\|.*<br>*Output example:*<br>*[KEY_SENTENCE_1] \|\|\|\|\| #Topic#*<br>*[KEY_SENTENCE_2] \|\|\|\|\| *Evidence** |
| Scenario: *OneDoc*<br>Task: *Extract* | **There are several different types of KEY SENTENCE in the input text, which are marked by special tags. These special tags a total of six kinds, respectively is <#Topic#>, <@argument@>, "<!Transition!>", "<\|Summary\|>", "<*Evidence*>", "<-Concession->". Different tags represent different types of key sentence. If a sentence in the text is KEY SENTENCE, we will add a special tag with the same attribute to the beginning and end of the sentence. The head tag also contains id order information in the format <type-id>. For example, the head tag with type '#Topic#' and id 1 is <#Topic#-1>. Also note that when the head tag and tail tag attributes are inconsistent, this means that the sentence is a fake KEY SENTENCE. Please read the input text carefully and give the answer directly according to the instruction requirements.**<br>Input text: Remember the essays you had to write in high school? Topic sentence, introductory paragraph, supporting paragraphs, conclusion. <#Topic#-2>With the result that writing is made to seem boring and pointless.<#Topic#> Who cares about symbolism in Dickens? ...<br>*Instruction: Gather every instance of KEY SENTENCE classified as #Topic#. The output should be a Json list arranged by ids. If none are found, provide an empty array.*<br>*Output Example 1: [KEY SENTENCE1, KEY SENTENCE2, KEY SENTENCE3,...]*<br>*Output Example 2: []* |
| Scenario: *OneDoc*<br>Task: *QA* | **There are several different types of KEY SENTENCE in the input text, which are marked by special tags. These special tags a total of six kinds, respectively is <#Topic#>, <@argument@>, "<!Transition!>", "<\|Summary\|>", "<*Evidence*>", "<-Concession->". Different tags represent different types of key sentence. If a sentence in the text is KEY SENTENCE, we will add a special tag with the same attribute to the beginning and end of the sentence. The head tag also contains id order information in the format <type-id>. For example, the head tag with type '#Topic#' and id 1 is <#Topic#-1>. Also note that when the head tag and tail tag attributes are inconsistent, this means that the sentence is a fake KEY SENTENCE. Please read the input text carefully and give the answer directly according to the instruction requirements.**<br>Input text: Remember the essays you had to write in high school? Topic sentence, introductory paragraph, supporting paragraphs, conclusion. <#Topic#-2>With the result that writing is made to seem boring and pointless.<#Topic#> Who cares about symbolism in Dickens? ...<br>*Instruction: Is "With the result that writing is made to seem boring and pointless." the KEY SENTENCE of the text? Indicate "True" for yes, "False" for no.* |

Table 12: Examples of all tasks in LIFBENCH. **Bold** denotes the scenario description $D$; normal denotes the context $X$; *italics* denotes instruction $I$, with red indicate the instruction variables $var$, and the remaining black parts correspond to the instruction template $tpl$.