Can Large Language Models Understand Real-World Complex Instructions?

Qianyu He[1], Jie Zeng[1], Wenhao Huang[1], Lina Chen[2], Jin Xiao[2], Qianxi He[1], Xunzhe Zhou[1], Jiaqing Liang[2*], Yanghua Xiao[1,3*]

[1]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
[2]School of Data Science, Fudan University
[3]Fudan-Aishu Cognitive Intelligence Joint Research Center, Shanghai, China
{qyhe21, jzeng23, whhuang21, lnchen23, jinxiao23, qxhe23}@m.fudan.edu.cn,
{xzzhou20, liangjiaqing, shawyh}@fudan.edu.cn

Abstract

Large language models (LLMs) can understand human instructions, showing their potential for pragmatic applications beyond traditional NLP tasks. However, they still struggle with complex instructions, which can be either complex task descriptions that require multiple tasks and constraints, or complex input that contains long context, noise, heterogeneous information and multi-turn format. Due to these features, LLMs often ignore semantic constraints from task descriptions, generate incorrect formats, violate length or sample count constraints, and be unfaithful to the input text. Existing benchmarks are insufficient to assess LLMs' ability to understand complex instructions, as they are close-ended and simple. To bridge this gap, we propose CELLO, a benchmark for evaluating LLMs' ability to follow complex instructions systematically. We design eight features for complex instructions and construct a comprehensive evaluation dataset from real-world scenarios. We also establish four criteria and develop corresponding metrics, as current ones are inadequate, biased or too strict and coarse-grained. We compare the performance of representative Chinese-oriented and English-oriented models in following complex instructions through extensive experiments. Resources of CELLO are publicly available at https://github.com/Abbey4799/CELLO.

Introduction

The emergence of large-scale models (Brown et al. 2020; Chowdhery et al. 2022; Touvron et al. 2023) has yielded noteworthy transformations in real-world applications (Richards 2023; Liu et al. 2023b). These models are able to understand a wide range of human instructions, spanning from casual conversations (Taori et al. 2023) to complex problems solving (Brown et al. 2020). Since human instructions are massive and diverse, traditional academic benchmarks that focus on specific tasks are no longer sufficient to evaluate LLMs (Zhong et al. 2023; Chia et al. 2023).

Real-world applications often involve a diverse range of complex instructions that significantly differ from the simple and common instructions in current benchmarks (Hendrycks et al. 2020; Huang et al. 2023), as shown in Fig. 1. Instruction generally consists of two parts (Honovich et al. 2022):

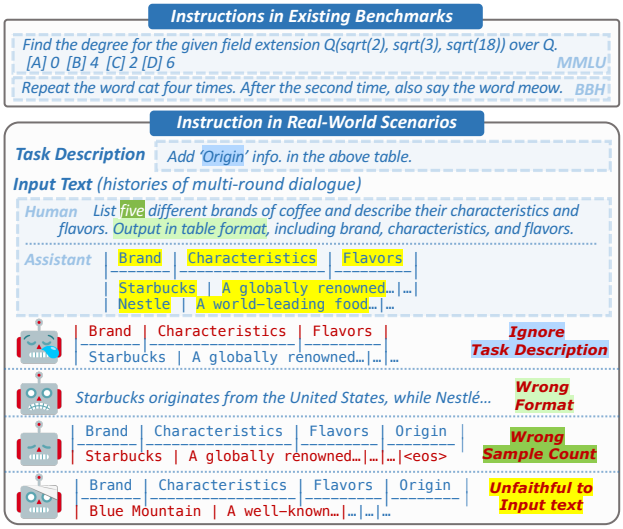

Figure 1: Existing benchmarks generally contain simple and common instructions. However, the complex instructions in real-world scenarios are a composition of multiple features, such as constraints on the output format, number of output samples, key elements of the output, and heterogeneity of input texts in the given example. The understanding of complex instructions poses challenges to current models.

Task description (mandatory) describes the task goal and *Input text* (optional) provides reference texts for the model to answer questions or the history of multi-turn conversations, as shown in Fig. 1. Hence, there can be two categories of complex instructions: *complex task descriptions* and *complex input*. Regarding *complex task descriptions*, models need to undertake multiple tasks (i.e. multi-tasking) and there can be diverse restrictions describing the task, including semantics constraints (e.g. the inclusion of key elements (Zhou et al. 2023a) or the use of predefined callable functions (Liu et al. 2023b)), format constraints (e.g. the predefined format in few-shot scenarios (Yao et al. 2023b) or structured format imitating human reasoning processes (Liu et al. 2023b)), quantity constraints (e.g. word, sentence, or sample count regulating the length of model output (Zhou et al. 2023b; Yao et al. 2023a)). Regarding *complex input*,

*Corresponding author.

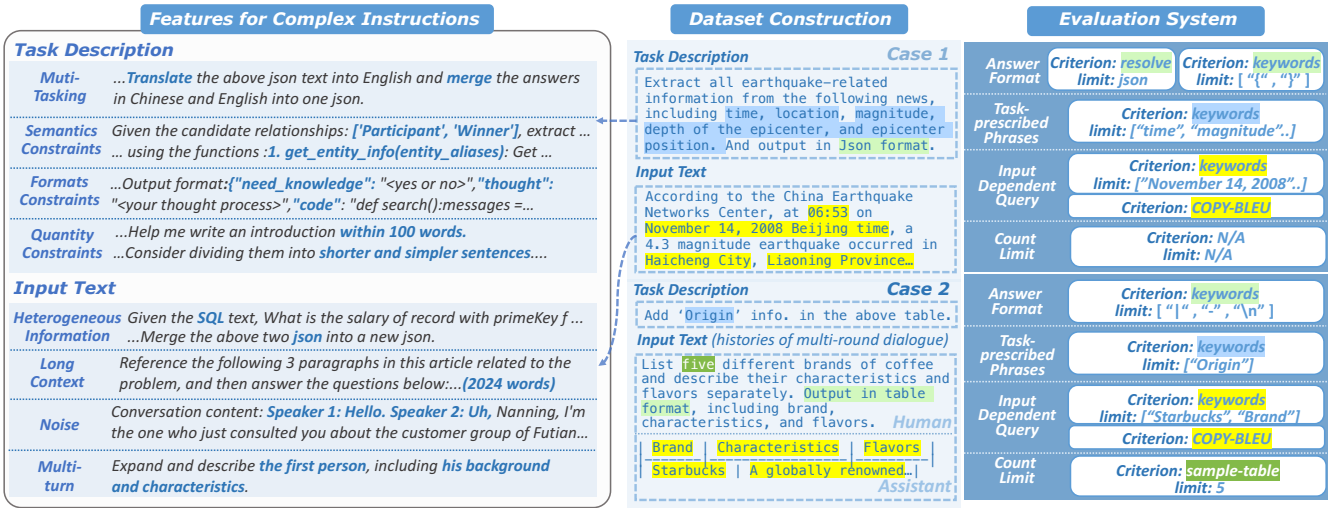

Figure 2: The framework of our benchmark design contains eight features for complex instructions, an evaluation dataset covering nine tasks, and four evaluation criteria along with their corresponding metrics.

the input text generally have long context (An et al. 2023; Liu et al. 2023a), noise (e.g. colloquial expressions (Guo et al. 2023) and error accumulation caused by pipeline method (Sun et al. 2023)), heterogeneous information (e.g. a combination of structured and unstructured data (Zha et al. 2023)), and in the form of multi-turn (Ding et al. 2023).

The complexity of real-world instructions accounts for prevalent errors observed in LLMs. As shown in Fig. 1, LLMs may (1) ignore semantic constraints from task description(s) (Zhou et al. 2023a), (2) generate answers in incorrect format (Qin et al. 2023), or (3) violate the length or sample count constraints (Zhou et al. 2023b), especially when multiple tasks are required to be performed. Moreover, models can (4) be unfaithful to the input text, especially when it is long, noisy, heterogeneous or in the form of multi-turn (Li et al. 2023b; An et al. 2023). Overall, complex instructions pose challenges to current models.

However, existing benchmarks are insufficient for effectively assessing the ability of LLMs to understand complex instructions. On one hand, Fig. 1 shows that existing benchmarks are either close-ended (Huang et al. 2023; Zhong et al. 2023; Yu et al. 2023) or contain common and simple instructions (Srivastava et al. 2023; Chia et al. 2023; Dubois et al. 2023), which fail to mirror the complexity of real-world instructions. On the other hand, even though certain benchmarks cover some of the above features of complex instructions, such as count restriction (Zhou et al. 2023b; Yao et al. 2023a), semantic restriction (Chen et al. 2022), and long text understanding (An et al. 2023), they only encompass isolated features, while real-world instructions comprehensively cover these features (Zhou et al. 2023a). Overall, none of the existing benchmarks systematically study the complex instructions understanding ability of LLMs.

In this paper, we propose CELLO, a benchmark for evaluating the **C**ompl**E**x instruction understanding ability of **L**arge **L**anguage M**O**dels systematically. The framework of our benchmark is shown in Fig. 2. As existing benchmarks only cover isolated features of complex instructions, we establish a comprehensive framework comprising eight features of complex instructions. Accordingly, we propose a novel evaluation system comprised of four criteria along with their corresponding metrics. The current evaluation criteria are insufficient to comprehensively reflect the ability of LLMs to understand complex instructions for the following reasons. First, complex instructions in real-world scenarios are open-ended (Xu et al. 2023), thus the criteria commonly used for close-ended benchmarks are not suitable in such cases (Hendrycks et al. 2020). Moreover, many studies adopt GPT4 evaluation for automated open-ended assessment, which introduces bias problems (Wang et al. 2023). Furthermore, the binary pass rate adopted by the benchmarks containing complex instructions is strict and coarse-grained, resulting in universally low scores for smaller LLM without discrimination (Liu et al. 2023b; Qin et al. 2023).

Overall, our contributions are mainly four-fold:

- To the best of our knowledge, we are the first to systematically investigate the ability of LLMs to follow complex instructions. We propose a comprehensive set of features for complex instructions, facilitating both dataset construction and evaluation criteria design.
- We construct a complex instruction dataset from real-world scenarios, containing 523 samples encompassing nine tasks, effectively covering our specified features. Specifically, we propose a two-stage framework for constructing the evaluation dataset for LLM's complex instruction understanding.
- We design four evaluation criteria and corresponding automatic metrics for assessing LLMs' ability to understand complex instructions in a comprehensive and discriminative way.
- We compare 19 representative Chinese-oriented models and 15 representative English-oriented models' performance on our benchmark.

Related Work

Evaluation for LLMs Many benchmarks propose comprehensive evaluation frameworks that integrate existing evaluation datasets (Liang et al. 2022; Zhong et al. 2023; Dubois et al. 2023; Chia et al. 2023). Mainstream benchmarks primarily focus on assessing knowledge (Huang et al. 2023; Gu et al. 2023; Yu et al. 2023), programming (Chen et al. 2021), and complex reasoning (Cobbe et al. 2021; Srivastava et al. 2023). Recently, many benchmarks focus on specific capabilities of models (Qin et al. 2023; Liu et al. 2023b; An et al. 2023). However, none of the existing benchmarks systematically investigate the ability of LLMs to follow complex instructions. Their evaluation criteria have several limitations when evaluating complex instruction understanding. First, the close-ended benchmarks fail to mirror the complexity of the real-world instructions (Huang et al. 2023; Gu et al. 2023; Zhong et al. 2023). Also, the binary success rate (Chen et al. 2021; Qin et al. 2023; Liu et al. 2023b) is too strict and coarse-grained, resulting in weak discrimination. Moreover, GPT-4 automatic scoring introduces bias problems (Wang et al. 2023). Overall, the existing benchmarks and their criteria are insufficient to effectively assess LLMs' ability to understand complex instructions.

Complex Instruction Following The current datasets generally have simple and common instructions, making LLMs challenging to follow complex instructions in real-world scenarios (Zhou et al. 2023a; Xu et al. 2023). Various methods have been proposed to improve models' understanding of complex instructions (Luo et al. 2023; Zhou et al. 2023a; Mukherjee et al. 2023). Despite the advancements, there is a lack of a benchmark for systematically evaluating models' understanding of complex instructions.

Evaluation for Constrained Instructions Many studies investigate the ability of LLMs to understand constrained instructions (Yao et al. 2023a; Zhou et al. 2023b; Chen et al. 2022). However, the instructions of these benchmarks are simplistic, and the constraints they involve are narrow.

CELLO Benchmark

As shown in Fig. 2, we first establish a framework containing eight features for complex instructions, then construct an evaluation dataset, and finally propose four evaluation criteria along with their corresponding metrics.

Dataset Construction

We first collect data from real scenarios, covering 9 tasks. Then we diversify the collected complex instructions through *In-breadth Evolution* and complicate the collected simple instructions through *In-breadth Evolution*.

Data Source and Selected Tasks When constructing the dataset, we take into account its *coverage* and *representativeness*. Regarding *coverage*, we include common NLP tasks found in existing benchmarks (Liang et al. 2022), while incorporating instructions with more complex task descriptions or input beyond those benchmarks. Moreover, we introduce specific tasks involving complex instructions, which align with common real-world applications for LLMs. Regarding *representativeness*, instructions are gathered from 90,000 user interaction logs over six months with our implemented chatbot. Finally, we include nine tasks, classified into six categories:

Complex NLP Tasks. Instructions concerning NLP tasks in real-world scenarios are more diverse and detailed (Xu et al. 2023) and contain noisy and long contexts (An et al. 2023) compared to academic datasets. Overall, we choose four tasks commonly found in existing benchmarks (Liang et al. 2022), enhancing them with more complex instructions and inputs beyond traditional benchmarks: *long text summarization*, *long text closed-domain question answering*, *long text keywords extraction*, *complex information extraction*. The details can be found in the Appendix.

Meta-prompt. Researchers design elaborate prompts to leverage LLMs to construct datasets (Xu et al. 2023; Honovich et al. 2022; Qin et al. 2023), which can be defined as *Meta-prompts* (Honovich et al. 2022). These prompts generally have varied instructions, rich input topics, few-shot samples, clear format requirements and are unlikely to appear in the training samples. Therefore, we collect prompts crafted by domain experts who focus on various real-world applications of LLMs, such as financial numerical reasoning and educational knowledge graph taxonomy construction, due to their high quality and origin in real-world scenarios.

Planning. Many studies have designed prompts to mimic human thinking processes, guiding LLMs to perform reasoning and planning (Yao et al. 2023b; Liu et al. 2023b). These prompts often impose restrictions on callable functions, have clear format requirements, offer few-shot samples, and provide long contexts. Therefore, we collect prompts that require LLMs to complete planning tasks based on CN-DBpedia (Xu et al. 2017), fund knowledge base, and those from Langchain[1]. Since smaller LLMs have limited planning capabilities (Liu et al. 2023b), we solely evaluate the models' ability to perform single-step planning.

Structured Input. Structured text is a common and crucial type of user input, due to its well-organized and easily interpretable format. Therefore, we include instructions with: (1) Six structured data types, namely Markdown, LaTeX, SQL, Tree, Python, JSON. (2) Two distinct tasks for their *complexity* and *representativeness*: *Path Compose* directly evaluates the model's understanding of complex nested data structures, while *TextRetrieval* is a common application to extract content meeting specific requirements. (3) Two levels of difficulty, which are categorized based on the length and depth of the structured input.

Well-guided Writing. Existing benchmarks (Chia et al. 2023) considering writing ability mainly have the following limitations: (1) They overlook the specific needs users have in real-world scenarios when seeking efficient writing guidance, such as word count, key information, or included hashtags. (2) They fail to consider the iterative nature of user satisfaction, as users may continually provide modification feedback. (3) They are difficult to automatically evaluate. To address these limitations, we collect various single-turn complex instructions covering various complex features and

[1] https://www.langchain.com/

Category	Tasks	#Samples	#Format	#Task	#Input	#Count	Avg TD Len.	Avg IP Len.	Avg Ins Len.
Complex Task Description	**Extraction**	49	49	35	49	N/A	125	169	295
	Planning	52	52	46	48	N/A	1070	534	1606
	Meta.	20	20	15	6	2	765	166	933
	BS(S)	20	20	20	1	15	70	N/A	70
	Writing(S)	23	2	23	2	12	82	25	107
Complex Input	**Keywords**	15	15	15	15	N/A	546	943	1579
	QA	89	N/A	N/A	89	N/A	25	881	814
	Sum.	108	N/A	N/A	108	N/A	45	514	562
	Structure	38	6	N/A	38	N/A	29	1360	1390
	BS(M)	52	50	50	10	36	31	559	31
	Writing(M)	57	3	35	48	43	30	656	51
Overall		523	217	239	414	108	256	528	676

Table 1: The statistics of our benchmark. For each task, #Format, #Task, #Input, #Count denote the number of samples covering the corresponding criteria. Avg TD/IP/Ins Len. denote the average word number of *task description*, *input text* and *instruction*. Meta., BS, SUM. denote the Meta-prompt, Brainstorming, Summarization task respectively. (S) and (M) represent single-round and multi-round. N/A denotes that such tasks do not involve corresponding evaluation criteria.

multi-turn instructions that reflect realistic revision needs.

Detailed Brainstorming. Brainstorming yields an intuitive impression for the chat models. However, existing evaluation datasets either have overly simple and open instructions that are difficult to evaluate (Li et al. 2023a), or they are excessively tricky with limited discrimination[2]. In our benchmark, we collect single-turn brainstorming data with detailed requirements and multi-turn brainstorming data that simulate realistic user interactions.

Data Evolution The collected complex instructions have two limitations: (1) For those collected from real-world projects, the human-elaborated task descriptions are complex but alike. (2) For those collected from usage logs, many simple instructions are not effectively utilized. Hence, we introduce two perspectives to evolve data, thereby achieving a more robust and reliable evaluation. **In-breadth Evolution** aims to diversify the collected complex instructions (including three methods *task description relocation*, *task description paraphrasing* and *task emulation*). **In-depth Evolution** aims to complicate the simple instructions to increase the data scale (including two methods *constraints addition*, *multi-round interaction*). The motivation and prompts for each method are detailed in the Appendix.

Evaluation System

Criteria We define the following criteria that should be assessed as they can encompass common errors made by models. (1) **Count limit**: the number of words, sentences, or samples allowed in the response. (2) **Answer format**: the expected structure or format of the response, such as a parsable JSON format, or a specified format for few-shot samples. (3) **Task-prescribed phrases**: semantic constraints on the response that are stipulated in the task description, such as predefined functions, primary subjects, or key elements. (4) **Input-dependent query**: the query should be answered faithfully according to the given input texts.

[2]https://github.com/zhenbench/z-bench

Although *Task-prescribed phrases* and *Input-dependent query* both impose content-related constraints on the response, they differ in the information they rely on. The former centers on constraints explicitly stated by the user in the task description, while the latter focuses on constraints implicitly derived from the content of the input text.

Evaluation Metrics We propose automated evaluation metrics for designed criteria, considering various perspectives and difficulty levels. Each sample $s_i = \{I_i, a_i, h_i\}$ consists of instruction I_i, a model answer a_i and given histories[3] $h_i = \{(I_0, a'_0), ..., (I_{i-1}, a'_{i-1})\}$. Here, i denotes the round number within multi-turn dialogues. For each sample s, its score for each criteria comprises multiple sub-scores $\mathcal{C} = \{c_1, c_2, ..., c_i\}$. Each sub-score $c_i = f_x(l, a_i, h_i)$ is determined by scoring function f_n based on the criterion x, and a limit l manually annotated by humans. The limit l can be an integer, a list of keywords, or a referenced string[4].

Count Limit. We mainly consider four sub-scores: *word count score*, *sentence count score*, and *sample count score*, *revise score*. For *word count score*, the criteria can be *word-max* and *word-min*. For the scoring function $f_{\text{word-max}}$, the more word count exceeds the threshold limit l_c, the lower the score will be, thus $f_{\text{word-max}}$ is defined as follows:

$$f_{\text{word-max}}(a_i, l_c) = \begin{cases} 1 & n(a_i) \leqslant l_c \\ 1 - \frac{|n(a_i) - l|}{n(a_i)} & n(a_i) > l_c \end{cases}$$

Here, $n(a_i)$ is the valid word count of answer a_i excluding punctuation marks. $f_{\text{word-min}}$ is defined as follows:

$$f_{\text{word-min}}(a_i, l_c) = \begin{cases} 1 & n(a_i) \geqslant l_c \\ \frac{n(a_i)}{l} & n(a_i) < l_c \end{cases}$$

Likewise, the scoring functions for *sentence count* encompass $f_{\text{sentence-max}}$, $f_{\text{sentence-min}}$, $f_{\text{sentence-exact}}$. The scoring

[3]To ensure a fair comparison between models, all the model answers in the histories for each sample are the same and provided by GPT-3.5-turbo.

[4]The annotation process is detailed in the Appendix.

Benchmark	Avg Ins Len.	Format	Metric	Obj.
C-Eval	110	C	ACC	T
AGIEval	184	C	EM/F1	T
WizardLM Testset	62	O	Preference	F
ToolBench	N/A	O	Pass Rate	T
			Preference	F
AgentBench	N/A	O	Pass Rate	T
CELLO	676	O	Four Fine-grained Metrics	T

Table 2: Statistics of existing benchmarks. Avg Ins denotes the average word numbers in instructions. C and O denote the Close-ended and Open-ended respectively. Preference refers to evaluation via GPT4. Obj. represents whether the evaluation metrics are objective (T) or subjective (F).

function for *sample count* $f_{\text{sample-exact}}$ is implemented using regex matching. The limit l_c for revise score f_{revise} can be the string *longer* or *shorter*. Speicifically, the function $f_{\text{revise}}(a_i, \textit{longer})$ equals 1 if $n(a_i) > n(a_{i-1})$, otherwise, it equals 0. For each sample, the final *Count Limit* score S_c is the average of all the sub-scores.

Answer Format. This metric has two sub-scores: *parseability* and *keywords*. First, if the model output can be parsed in the prescribed format, such as JSON, $f_{\text{parseability}}(a_i, \textit{json})$ equals 1; otherwise, it equals 0. However, even in cases where the model output cannot be directly parsed, its ability to learn certain patterns still demonstrates its capacity to follow complex instructions. Consequently, for each sample, we first extract keywords list $l_f = \{w_1, w_2, ..., w_i\}$ from pre-defined formats, which we define as *Scoring Keywords*. Then, the sub-score $f_{\text{keywords}}(a_i, l_f)$ is defined as follows:

$$f_{\text{keywords}}(a_i, l_f) = \frac{N(a_i, l_f)}{|l_f|},$$

where N denotes the number of scoring keywords covered by the model output a_i. Finally, the overall score for answer format S_f is the average of $f_{\text{parseability}}$ and f_{keywords}.

Input-dependent Query. The key phrases of the correct answer stem from the input text. The more scoring keywords included in a response, the higher the quality of the response. Hence, for each sample, the subscore $f_{\text{keywords}}(a_i, l)$ is also applied here, where the *Scoring keywords* l_q are extracted from *input text*. Moreover, certain models tend to repeat input text when they fail to understand the instructions, especially when the input text is long and noisy or during the multi-turn dialogue. To prevent this undesirable copying behavior, we introduce a penalty term known as COPY-BLEU (Chen et al. 2022), which decreases as the response exhibits greater similarity to the input text. The final score S_q for the Input-dependent query is defined as follows:

$$S_q = (1 - f_{\text{BLEU}}(a_i, t_i)) f_{\text{keywords}}(a_i, l_q),$$

where t_i is the input text of sample s_i.

Task-prescribed Phrases. The mandatory phrases specified in the task description are essential conditions that must be fulfilled. The more mandatory phrases covered in the answers, the better the model follows complex instructions. Hence, the subscore $f_{\text{keywords}}(a_i, l_t)$ is applied where l_t is the scoring keywords extracted from the task description.

Evaluation of the Benchmark

Each sample is labeled by three annotators. Specifically, we retain samples only when at least two annotators agree on the criteria *Count Limit* and *Output Format Parseability*. For criteria involving *Keywords Coverage*, we only keep keywords with a consensus from at least two annotators.

Statistics of the Benchmark

Tab. 1 presents the statistics[5] of CELLO. Our dataset has two categories depending on whether the criteria are mainly in the task description or the input text. Different tasks also have different emphases on the criteria, and our dataset covers the four criteria effectively. Tab. 2 compares our benchmark with existing ones. Our benchmark is the first to systematically test LLMs' ability to follow complex instructions, which are generally longer and more complex than other benchmarks. The tasks we cover are open-ended, which are more realistic and practical. Our evaluation is also more objective and fine-grained.

Experiment

Evaluated Models We evaluate a total of 34 models that demonstrated exceptional performance on other benchmarks (Huang et al. 2023; Dubois et al. 2023), ranging from their model size, supported context length, and instruction tuning data size, as illustrated in Appendix. These models are categorized into three groups: Chinese-oriented Models *(From Scratch, FS)*, Chinese-oriented Models *(Continue Pretraining, CP)*, and English-oriented Models. The distinction between English and Chinese-oriented Models lies in the composition of their pretraining corpus, whereby the former possesses a small portion and the latter possesses a substantial volume of Chinese data. Chinese-oriented Models *(FS)* are trained entirely from scratch using Chinese corpora. Chinese-oriented Models *(CP)* continue pretraining on Chinese corpora utilizing an English-oriented base model.

Task-categorized Performance The performance of the models on different tasks is shown in Tab. 3.

General Comparisons. Among the models assessed, OpenChat-V3.2 was the best, followed by Vicuna-V1.5-13B and ChatGLM. These models had different parameter sizes, showing that small-scale LLMs can follow complex instructions as well as larger ones. The Chinese-oriented (FS) group and the English-oriented group perform equally

[5]Chinese word are counted via https://github.com/fxsjy/jieba. English words are counted via https://www.nltk.org/.

Model	Complex Task Description					Complex Input						All
	Extract.	Plan.	Meta.	Wri.(S)	BS.(S)	Key.	QA.	Sum.	Struct.	Wri.(M)	BS.(M)	Avg.
Chinese-oriented Models (Continue Pretraining)												
Baize-V2-7B	0.203	0.266	0.300	0.504	0.245	0.056	0.121	0.045	0.593	0.381	0.558	0.298
Llama2-FlagAlpha	0.205	0.095	0.129	0.262	0.547	0.150	0.423	0.297	0.354	0.406	0.591	0.309
Baize-V2-13B	0.214	0.334	0.342	0.272	0.536	0.070	0.143	0.019	0.540	0.433	0.574	0.318
Alpaca-V1-13B	0.289	0.183	0.209	0.209	0.697	0.411	0.272	0.226	0.399	0.291	0.480	0.332
Alpaca-V1-7B	0.264	0.123	0.215	0.357	0.612	0.265	0.267	0.243	0.465	0.401	0.703	0.352
Llama2-Linly	0.382	0.170	0.205	0.352	0.527	0.196	0.464	0.406	0.596	0.352	0.594	0.381
Alpaca-V1-33B	0.379	0.200	0.283	0.664	0.663	0.415	0.334	0.221	0.426	0.476	0.609	0.426
BELLE	0.400	0.157	0.363	0.589	0.734	0.379	0.478	0.508	0.458	0.439	0.672	0.469
CuteGPT	0.482	0.529	0.460	0.534	0.739	0.294	0.506	0.459	0.653	0.626	0.804	0.553
Llama2-LinkSoul	0.521	0.326	0.431	0.652	0.769	0.615	**0.788**	0.684	0.565	0.747	0.909	0.629
Llama2-OpenBuddy	0.585	0.638	0.344	0.697	0.697	0.638	0.752	0.685	*0.711*	**0.812**	0.892	0.670
Chinese-oriented Models (From Scratch)												
BatGPT-sirius	0.011	0.044	0.094	0.352	0.233	0.046	0.394	0.054	0.294	0.135	0.321	0.177
MOSS	0.493	0.310	0.461	0.634	0.644	0.473	0.396	0.500	0.521	0.696	0.658	0.525
InternLM	0.452	0.540	0.493	0.690	0.622	0.247	0.515	0.399	0.428	0.732	0.877	0.546
ChatGLM2	0.539	0.317	*0.608*	0.664	0.632	0.589	0.725	0.669	0.590	0.738	0.777	0.616
ChatGLM2-32k	0.526	0.399	0.572	0.699	0.690	0.653	0.686	0.571	0.427	0.758	0.876	0.620
Baichuan-chat	0.473	0.373	0.471	**0.800**	0.794	0.491	0.728	*0.701*	0.601	0.776	0.857	0.637
Qwen	0.544	0.551	0.493	0.646	0.740	0.486	*0.767*	0.705	0.575	0.710	0.888	0.642
ChatGLM	**0.649**	0.522	0.612	0.700	0.808	0.532	0.742	0.672	0.573	0.735	0.870	*0.673*
English-oriented Models												
Llama2-chat-7B	0.495	0.326	0.500	0.358	0.465	0.157	0.135	0.060	0.708	0.541	0.447	0.385
Llama2-chat-70B	0.431	0.289	0.484	0.397	0.472	0.147	0.158	0.079	0.719	0.570	0.552	0.393
Llama2-chat-13B	0.445	0.329	**0.624**	0.359	0.453	0.154	0.127	0.108	**0.753**	0.569	0.458	0.402
Vicuna-V1.3-7B	0.485	0.661	0.303	0.748	0.665	0.180	0.651	0.583	0.525	0.674	0.773	0.569
WizardLM	0.422	0.592	0.281	0.675	0.565	0.261	0.594	0.570	0.519	0.711	0.839	0.574
LongChat-V1-13B	0.523	0.591	0.423	0.654	0.533	0.400	0.572	0.532	0.579	0.752	0.810	0.576
LongChat-V1.5-7B	0.489	0.620	0.358	0.664	0.731	0.608	0.687	0.633	0.378	0.747	0.825	0.609
LongChat-V1-7B	0.549	0.475	0.424	0.710	0.805	0.527	0.604	0.557	0.692	0.729	0.856	0.627
Vicuna-V1.3-13B	0.521	0.625	0.474	0.743	*0.840*	0.346	0.672	0.582	0.613	0.651	0.869	0.631
Vicuna-V1.5-7B	0.544	0.670	0.398	0.506	0.770	0.711	0.739	0.667	0.513	0.693	*0.906*	0.641
Vicuna-V1.3-33B	0.589	*0.702*	0.385	*0.752*	0.835	0.503	0.680	0.643	0.627	0.622	0.872	0.655
Vicuna-V1.5-13B	*0.601*	0.721	0.425	0.744	0.794	*0.682*	0.765	**0.723**	0.630	0.746	0.896	0.699
OpenChat-V3.2	0.629	**0.733**	0.510	0.754	**0.868**	**0.725**	0.771	0.663	0.608	*0.761*	**0.919**	**0.720**
GPT-3.5-turbo	0.709	0.805	0.632	0.879	0.854	0.765	0.795	0.832	0.697	0.879	0.908	0.794
GPT-4	0.737	0.879	0.666	0.828	0.810	0.862	0.889	0.911	0.727	0.867	0.910	0.822

Table 3: The performance of models on different tasks. Detailed information of each model is provided in the Appendix. The bold, underlined, and italicized denote the first, second, and third rankings, respectively. Here, Extract., Plan., Meta., Key., Sum., Struct., Avg. denote Extraction, Planning, Meta-prompt, Keywords, Summarization, Structure, Average respectively.

well and better than the Chinese-oriented (CC) group, proving that complex instruction comprehension is not language-dependent. Moreover, under the same base model, vocabulary, and supported context length (e.g. Llama2-7B), the performance of the models varies greatly. This demonstrates a strong correlation between the ability to comprehend complex instructions and the instruction tuning phase. Overall, the current open-source small to medium-scale models exhibit a significant performance gap compared to close-source large-scale models (GPT-3.5-turbo, GPT4).

Complex Task Description. Among the data with complex task descriptions, first, four of the top 5 models belong to the English-oriented Models, which demonstrate that the ability to understand complex task descriptions can transfer across different languages. Next, within the same series of models, larger model sizes do not always lead to improvements. Furthermore, the best-performing models in each group have a supported context length of less than 4096, suggesting that the supported text context length does not significantly impact the ability to comprehend complex task descriptions.

Complex Input Text. For the data with complex input text, first, seven of the top 10 models belong to Chinese-oriented models, which implies that more Chinese training data assists the models in comprehending long and noisy Chinese texts. Next, within the same model series, larger scales generally improve performance, while longer supported context length can result in performance drops in many cases.

Criteria-categorized Performance As shown in Tab. 4, regarding *Answer format*, the English-oriented Models sig-

Model	Format	Input	Task	Count
Chinese-oriented Models (Continue Pretraining)				
Baize-V2-7B	0.409	0.300	0.246	0.466
Llama2-FlagAlpha	0.499	0.218	0.221	0.468
Baize-V2-13B	0.530	0.247	0.302	0.444
Alpaca-V1-13B	0.603	0.207	0.259	0.458
Alpaca-V1-7B	0.663	0.224	0.256	0.512
Llama2-Linly	0.411	0.347	0.374	0.490
Alpaca-V1-33B	0.655	0.353	0.357	0.576
BELLE	0.556	0.408	0.484	0.498
CuteGPT	0.640	0.548	0.576	0.514
Llama2-LinkSoul	0.662	0.623	0.662	0.603
Llama2-OpenBuddy	0.734	0.627	0.704	0.638
Chinese-oriented Models (From Scratch)				
BatGPT-sirius	0.154	0.206	0.069	0.357
MOSS	0.586	0.514	0.564	0.534
InternLM	0.650	0.527	0.524	0.612
ChatGLM2	0.620	0.605	0.691	0.568
ChatGLM2-32k	0.687	0.563	*0.716*	0.603
Baichuan-chat	0.750	0.603	0.586	*0.662*
Qwen	0.764	0.584	0.625	0.570
ChatGLM	0.715	*0.628*	0.742	0.571
English-oriented Models				
Llama2-chat-7B	0.598	0.294	0.306	0.686
Llama2-chat-70B	0.631	0.318	0.265	**0.701**
Llama2-chat-13B	0.640	0.342	0.280	0.674
Vicuna-V1.3-7B	0.598	0.520	0.599	0.597
WizardLM	0.730	0.525	0.531	0.586
LongChat-V1-13B	0.723	0.528	0.585	0.507
LongChat-V1.5-7B	**0.791**	0.518	0.589	0.535
LongChat-V1-7B	0.789	0.574	0.615	0.609
Vicuna-V1.3-13B	0.766	0.588	0.641	0.554
Vicuna-V1.5-7B	0.756	0.536	0.698	0.599
Vicuna-V1.3-33B	0.770	0.609	0.668	0.575
Vicuna-V1.5-13B	*0.786*	0.656	0.701	0.640
OpenChat-V3.2	0.766	**0.703**	**0.776**	0.617
GPT-3.5-turbo	0.899	0.760	0.799	0.700
GPT-4	0.911	0.796	0.792	0.724

Table 4: The performance of models for different criteria. The bold, underlined, and italicized denote the first, second, and third rankings, respectively.

nificantly perform better than Chinese-oriented Models. This demonstrates the English-oriented Models' ability to follow few-shot examples and generate code, as well as partially explains why their complex instruction-following ability can transfer across languages. Next, for *Task-prescribed phrases*, two of the top-3 models are Chinese-oriented Models, suggesting that Chinese data helps the models understand Chinese semantic restrictions. Finally, the performance differences between models for *Count limit* criteria are not big compared to other criteria, which shows that the models have similar comprehension of numerical concepts.

Comparisons between Benchmarks We present the performance[6] on mainstream benchmarks in Fig. 3. First, on benchmarks focusing on Chinese knowledge, smaller models achieve similar or even better performance compared to GPT-3.5-turbo. Also, on challenging benchmarks like complex reasoning and programming ability, there is a lack of

[6]https://opencompass.org.cn/leaderboard-llm.

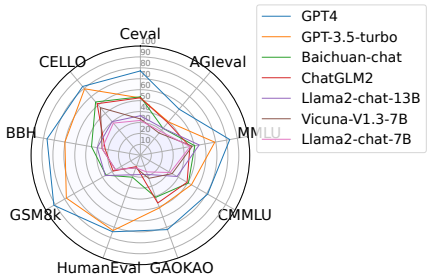

Figure 3: Model performance on mainstream benchmarks.

distinction between smaller models. Overall, our benchmark can exhibit more discriminative results.

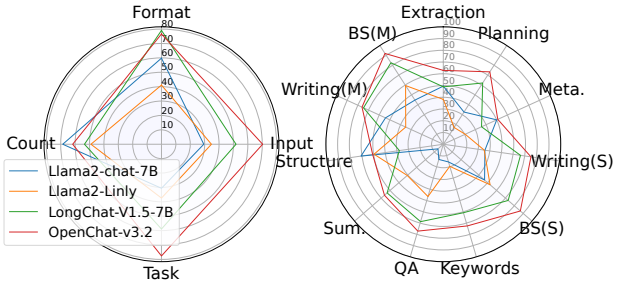

Figure 4: LLMs' performance on different tasks and criteria based on the same model (Touvron et al. 2023)

Fine-grained Evaluation Fig. 4 shows the performance of LLMs based on the same base model for different tasks and criteria. Different models have different strengths for different criteria. For example, Llama2-chat-7B is good at understanding format but bad at comprehending Chinese input and semantic constraints. Different models also excel in specific tasks. Llama2-chat-7B handles complex task descriptions well, but not complex input text.

Conclusion

In this work, we systematically investigate the complex instructions following ability of LLMs. We establish a framework comprising eight features for complex instructions, then construct an evaluation dataset covering nine tasks, and finally propose four evaluation criteria and corresponding metrics to assess LLMs' complex instruction understanding ability. Furthermore, we conduct extensive experiments to compare the performance of representative models.

Acknowledgements

This work is supported by Science and Technology Commission of Shanghai Municipality Grant (No. 22511105902), National Natural Science Foundation of China (No.62102095), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). Yanghua Xiao is also a member of Research Group of Computational and AI Communication at Institute for Global Communications and Integrated Media, Fudan University.

References

An, C.; Gong, S.; Zhong, M.; Li, M.; Zhang, J.; Kong, L.; and Qiu, X. 2023. L-Eval: Instituting Standardized Evaluation for Long Context Language Models. *arXiv preprint arXiv:2307.11088*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, H.; Li, H.; Chen, D.; and Narasimhan, K. 2022. Controllable Text Generation with Language Constraints. *arXiv preprint arXiv:2212.10466*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. d. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chia, Y. K.; Hong, P.; Bing, L.; and Poria, S. 2023. INSTRUCTEVAL: Towards Holistic Evaluation of Instruction-Tuned Large Language Models. *arXiv preprint arXiv:2306.04757*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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

Ding, N.; Chen, Y.; Xu, B.; Qin, Y.; Zheng, Z.; Hu, S.; Liu, Z.; Sun, M.; and Zhou, B. 2023. Enhancing Chat Language Models by Scaling High-quality Instructional Conversations. *arXiv preprint arXiv:2305.14233*.

Dubois, Y.; Li, X.; Taori, R.; Zhang, T.; Gulrajani, I.; Ba, J.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. *arXiv preprint arXiv:2305.14387*.

Gu, Z.; Zhu, X.; Ye, H.; Zhang, L.; Wang, J.; Jiang, S.; Xiong, Z.; Li, Z.; He, Q.; Xu, R.; et al. 2023. Xiezhi: An Ever-Updating Benchmark for Holistic Domain Knowledge Evaluation. *arXiv preprint arXiv:2306.05783*.

Guo, B.; Zhang, X.; Wang, Z.; Jiang, M.; Nie, J.; Ding, Y.; Yue, J.; and Wu, Y. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Honovich, O.; Scialom, T.; Levy, O.; and Schick, T. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*.

Huang, Y.; Bai, Y.; Zhu, Z.; Zhang, J.; Zhang, J.; Su, T.; Liu, J.; Lv, C.; Zhang, Y.; Lei, J.; et al. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *arXiv preprint arXiv:2305.08322*.

Li, G.; Hammoud, H. A. A. K.; Itani, H.; Khizbullin, D.; and Ghanem, B. 2023a. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*.

Li, J.; Cheng, X.; Zhao, W. X.; Nie, J.-Y.; and Wen, J.-R. 2023b. HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models. *arXiv e-prints*, arXiv–2305.

Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A.; et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Liu, N. F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; and Liang, P. 2023a. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; et al. 2023b. AgentBench: Evaluating LLMs as Agents. *arXiv preprint arXiv:2308.03688*.

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

Mukherjee, S.; Mitra, A.; Jawahar, G.; Agarwal, S.; Palangi, H.; and Awadallah, A. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Qin, Y.; Liang, S.; Ye, Y.; Zhu, K.; Yan, L.; Lu, Y.; Lin, Y.; Cong, X.; Tang, X.; Qian, B.; et al. 2023. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. *arXiv preprint arXiv:2307.16789*.

Richards, T. B. 2023. Auto-GPT: An Autonomous GPT-4 Experiment.

Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A. A. M.; Abid, A.; Fisch, A.; Brown, A. R.; Santoro, A.; Gupta, A.; Garriga-Alonso, A.; et al. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Sun, W.; Yan, L.; Ma, X.; Ren, P.; Yin, D.; and Ren, Z. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *arXiv preprint arXiv:2304.09542*.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford alpaca: An instruction-following llama model.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, P.; Li, L.; Chen, L.; Zhu, D.; Lin, B.; Cao, Y.; Liu, Q.; Liu, T.; and Sui, Z. 2023. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Xu, B.; Xu, Y.; Liang, J.; Xie, C.; Liang, B.; Cui, W.; and Xiao, Y. 2017. CN-DBpedia: A never-ending Chinese knowledge extraction system. In *International Conference*

on Industrial, Engineering and Other Applications of Applied Intelligent Systems, 428–438. Springer.

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

Yao, S.; Chen, H.; Hanjie, A. W.; Yang, R.; and Narasimhan, K. 2023a. COLLIE: Systematic Construction of Constrained Text Generation Tasks. *arXiv preprint arXiv:2307.08689*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023b. ReAct: Synergizing Reasoning and Acting in Language Models (arXiv: 2210.03629). arXiv.

Yu, J.; Wang, X.; Tu, S.; Cao, S.; Zhang-Li, D.; Lv, X.; Peng, H.; Yao, Z.; Zhang, X.; Li, H.; et al. 2023. KoLA: Carefully Benchmarking World Knowledge of Large Language Models. *arXiv preprint arXiv:2306.09296*.

Zha, L.; Zhou, J.; Li, L.; Wang, R.; Huang, Q.; Yang, S.; Yuan, J.; Su, C.; Li, X.; Su, A.; et al. 2023. TableGPT: Towards Unifying Tables, Nature Language and Commands into One GPT. *arXiv preprint arXiv:2307.08674*.

Zhong, W.; Cui, R.; Guo, Y.; Liang, Y.; Lu, S.; Wang, Y.; Saied, A.; Chen, W.; and Duan, N. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Zhou, C.; Liu, P.; Xu, P.; Iyer, S.; Sun, J.; Mao, Y.; Ma, X.; Efrat, A.; Yu, P.; Yu, L.; et al. 2023a. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

Zhou, W.; Jiang, Y. E.; Wilcox, E.; Cotterell, R.; and Sachan, M. 2023b. Controlled text generation with natural language instructions. *arXiv preprint arXiv:2304.14293*.