CodeScope: An Execution-based Multilingual Multitask Multidimensional Benchmark for Evaluating LLMs on Code Understanding and Generation

Weixiang Yan*, Haitian Liu*, Yunkun Wang*, Yunzhe Li*, Qian Chen, Wen Wang, Tingyu Lin, Weishan Zhao, Li Zhu, Hari Sundaram, Shuiguang Deng

*Equal contribution, corresponding to: weixiangyan@ucsb.edu

Challenges & Motivations

Existing benchmarks for evaluating the code understanding and generation capacities of LLMs suffer from severe limitations:

1. **Limited Language and Task Scope**

   Most benchmarks are insufficient as they focus on a narrow range of programming languages and specific tasks.

2. **Neglecting Executability and Consistency**

   Most benchmarks fail to consider the actual executability and the consistency of execution results of the generated code.

Benchmark	Execution-Based	Multilingual	Multitask	Multidimensional
HumanEval	✓	✗	✗	✗
MBPP	✓	✗	✗	✗
CodeXGlue	✗	✓(9)	✓(10)	✗
XLCoST	✗	✓(7)	✓(5)	✗
MathQA	✓	✗	✗	✗
MBXP	✓	✓(13)	✗	✗
ClassEval	✓	✗	✗	✗
MultiPL-E	✓	✓(18)	✗	✗
AiXBench	✓	✗	✗	✗
DS-1000	✓	✗	✗	✗
APPS	✓	✗	✗	✗
HumanEval-X	✓	✓(5)	✓(2)	✗
XCodeEval	✓	✓(11)	✓(5)	✗
CodeScope	✓	✓(43)	✓(8)	✓

Comparisons between CodeScope and existing code evaluation benchmarks.

Introduction & Contributions

CodeScope, a benchmark that evaluates the coding proficiency of LLMs using **execution-based** metrics in a **multilingual** and **multitask** setting. CodeScope consists of **eight tasks** for code understanding and generation, covering **43 programming languages**. We develop a multilingual code execution engine, **MultiCodeEngine**, which supports the compilation and execution of **14 programming languages**. Our contributions can be summarized as follows:

- **CodeScope benchmark**: We built the first-ever comprehensive benchmark for evaluating LLMs on code understanding and generation tasks.
- **Multidimensional fine-grained evaluation**: We comprehensively evaluate the performance of LLMs on eight tasks from three dimensions (length, difficulty, efficiency).
- **Comprehensive evaluations and in-depth analyses**: We evaluate the coding capabilities of eight mainstream LLMs and conduct comprehensive validations and analyses of the utility of the CodeScope benchmark.

Category	Dimension	Task	#Lang.	#Samples	Length
Understanding	Length	Code Summarization	43	4,838	385
		Code Smell	2	200	650
		Code Review	9	900	857
		Automated Testing	4	400	251
Generation	Difficulty	Program Synthesis	14	803	538
		Code Translation	14	5,382	513
		Code Repair	14	746	446
	Efficiency	Code Optimization	4	121	444

Summary of our CodeScope. #Lang. donates the number of languages covered by the task, **#Samples** donates the number of samples included in the task, and **Length** donates the average number of tokens per sample in the task.

Multidimensional Evaluation of LLMs' Coding Capabilities

***Length*-dimension** evaluation on code understanding tasks. SD means standard deviation.

Code Summarization

Model	Short	Medium	Long	Avg.	SD
GPT-4	33.78	33.27	33.88	**33.66**	0.33
GPT-3.5	33.21	32.87	33.51	33.14	**0.32**
Vicuna	32.12	32.21	31.62	32.06	**0.32**
WizardCoder	32.85	32.05	29.01	31.99	2.03
Code LLaMA	32.39	31.36	28.59	31.52	1.97
LLaMA 2	32.03	31.25	29.34	31.40	1.38
StarCoder	31.63	30.69	30.08	31.18	0.78
PaLM 2	31.83	29.95	24.20	30.27	3.98

Code Smell

Model	Short	Medium	Long	Avg.	SD
WizardCoder	45.09	48.29	53.03	**48.80**	**3.99**
LLaMA 2	41.13	31.77	49.28	40.73	8.76
Vicuna	38.94	30.66	39.54	36.38	4.96
GPT-4	30.44	40.02	37.60	36.02	4.98
PaLM 2	28.48	41.61	36.14	35.41	6.60
GPT-3.5	29.12	38.13	37.55	34.93	5.04
Code LLaMA	34.78	40.79	24.10	33.22	8.45
StarCoder	28.75	19.79	14.13	20.89	7.37

Code Review

Model	Short	Medium	Long	Avg.	SD
Code LLaMA	39.34	44.70	43.66	**42.57**	2.84
GPT-4	44.08	39.93	41.69	41.90	2.08
LLaMA 2	45.74	40.05	39.14	41.64	3.58
PaLM 2	41.56	42.13	39.79	41.16	**1.22**
Vicuna	43.92	38.70	40.43	41.02	2.66
GPT-3.5	45.75	37.88	34.56	39.40	5.75
WizardCoder	32.68	41.05	43.36	39.03	5.62
StarCoder	45.34	39.02	32.20	38.85	6.57

Automated Testing

Model	Short	Medium	Long	Avg.	SD
GPT-3.5	87.49	86.37	80.91	**84.92**	3.52
PaLM 2	84.52	81.97	80.38	82.29	2.09
LLaMA 2	83.46	80.48	80.27	81.40	1.78
Code LLaMA	82.65	79.34	80.27	80.75	**1.71**
WizardCoder	82.25	82.13	77.87	80.75	2.49
StarCoder	78.70	80.77	72.96	77.48	4.05
GPT-4	80.80	75.03	75.33	77.05	3.25
Vicuna	75.19	74.85	79.15	76.40	2.39

Length

Model	Overall	Avg.(SD)
WizardCoder	50.14	3.53
LLaMA 2	48.79	3.88
GPT-3.5	48.10	3.66
PaLM 2	47.28	3.47
GPT-4	47.16	2.66
Code LLaMA	47.02	3.74
Vicuna	46.47	2.68
StarCoder	42.10	4.69

***Difficulty*-dimension** evaluation on program synthesis, code translation, code repair tasks, evaluated using Pass@K and DSR@K testing.

Program Synthesis

Model	Easy	Hard	Avg.
GPT-4	58.57	12.01	**36.36**
GPT-3.5	39.29	4.96	22.91
Code LLaMA	7.14	0.26	3.86
WizardCoder	5.95	0.26	3.24
PaLM 2	3.81	0.78	1.99
LLaMA 2	1.43	0.00	0.75
StarCoder	0.95	0.00	0.50
Vicuna	0.71	0.00	0.37

Code Translation

Model	Easy	Hard	Avg.
GPT-4	40.26	22.06	**31.29**
GPT-3.5	28.50	14.03	21.37
WizardCoder	8.83	3.24	6.07
StarCoder	5.75	1.89	3.85
PaLM 2	5.27	1.70	3.51
Code LLaMA	4.91	1.66	3.31
LLaMA 2	1.10	0.26	0.69
Vicuna	0.62	0.19	0.41

Code Repair

Model	Easy	Hard	Avg.
GPT-4	43.56	14.04	**30.03**
GPT-3.5	18.56	7.60	13.54
PaLM 2	7.43	7.02	7.24
Wizardcoder	4.95	5.56	5.23
Code LLaMA	4.21	3.51	3.89
Vicuna	3.47	2.34	2.95
Starcoder	2.23	3.51	2.82
LLaMA 2	1.49	1.46	1.47

Difficulty

Model	Overall
GPT-4	**32.56**
GPT-3.5	19.27
WizardCoder	4.85
PaLM 2	4.25
Code LLaMA	3.68
StarCoder	2.39
Vicuna	1.24
LLaMA 2	0.97

***Efficiency*-dimension** evaluation on code optimization task, evaluated using Opt@K testing.

Model	Python		C		C++		C#		Overall
	Memory	Time	Memory	Time	Memory	Time	Memory	Time	
GPT-4	46.67	36.67	43.33	6.67	29.04	3.23	36.67	23.33	**28.20**
GPT-3.5	40.00	20.00	76.67	6.67	29.03	19.35	0.00	20.00	26.46
WizardCoder	50.00	16.67	50.00	0.00	38.71	12.90	10.00	16.67	24.37
Code LLaMA	43.33	13.33	40.00	0.00	35.48	3.22	10.00	23.33	21.09
PaLM 2	20.00	13.33	20.00	0.00	6.45	6.45	0.00	6.67	9.11
StarCoder	20.00	6.67	13.33	0.00	16.13	0.00	3.33	6.67	8.27
LLaMA 2	16.67	3.33	16.67	6.67	6.45	0.00	6.67	0.00	7.06
Vicuna	20.00	6.67	13.33	0.00	6.45	0.00	0.00	6.67	6.64

1. **Length-dimension**: **Short**, **Medium**, and **Long** are the length classifications of the code. **Code Summarization**, which requires the model to concisely summarize the core functionality and intent of the code; **Code Smell**, which requires the model to detect potential programming issues and poor practices in snippets within the input code; **Code Review**, which requires the model to evaluate the overall quality, style, and errors of the code; **Automated Testing**, which requires the model to understand the programming logic, data flow, and execution process of the code.
2. **Difficulty-dimension**: **Easy** and **Hard** categories refer to the difficulty. **Program Synthesis** (Correctness), which evaluates the ability of LLMs to generate correct code according to the given NL-description; **Code Translation** (Compatibility), which examines whether LLMs can maintain functional consistency when translating between different programming languages; **Code Repair** (Maintainability), which focuses on LLMs' ability to detect and fix errors automatically.
3. **Efficiency-dimension**: **Memory** and **Time** are two aspects for evaluating code efficiency. **Code Optimization**, which evaluates LLMs' capability to improve the performance and resource consumption of the code.

Results Analysis & Comparison

The CodeScope benchmark results can be viewed from three aspects: (a) Code Understanding capability; (b) Code Generaiton capability; (c) Overall coding capability.

CodeScope (Understanding)

- **We evaluate the LLMs' ability to interpret and analyze code.** We calculate the average performance of each model on four code understanding tasks, and use it as their overall score in this domain. The rankings of these LLMs in code understanding are different from their rankings in HumanEval and MBPP.
- For example, GPT-4, which ranks highest in HumanEval and MBPP, is only fifth in CodeScope (Understanding). This indicates that **strong performance in code generation tasks does not necessarily imply a good understanding of complex code.**

CodeScope (Generation)

- We use the same method to calculate the overall score. GPT-4 and GPT-3.5 do much better in code generation than in HumanEval and MBPP. This disparity may be attributed to two reasons:
- First, **CodeScope (Generation) tests the general ability of LLMs to generate code for multiple tasks and languages**. Unlike HumanEval and MBPP, which only test NL-to-PL tasks, CodeScope tests NL-to-PL, PL-to-PL, and NL&PL-to-PL tasks, examining the correctness, quality, and efficiency of the generated code, as well as the adaptability of LLMs to different languages.
- Second, **CodeScope (Generation) presents more complex and diverse problems, with varying levels of difficulty.** In contrast, HumanEval and MBPP feature simpler, predefined problems. For instance, the average number of tokens in solutions is 53.8 and 57.6 for HumanEval and MBPP, respectively, but 507.6 for CodeScope (Generation). Consequently, **some LLMs that perform well in HumanEval and MBPP, such as WizardCoder, do not fare as well in CodeScope (Generation).**

CodeScope (Overall)

- **The rankings of LLMs on CodeScope, HumanEval, and MBPP are not consistent.** This inconsistency highlights the advantages of CodeScope in terms of its breadth and challenge. CodeScope evaluates both code generation and code understanding skills, which are more relevant for real-world programming scenarios.
- **CodeScope** employs multilingual, multidimensional, multitask, and execution-based evaluation methods, **enhancing the difficulty and diversity of the evaluation.** CodeScope simulates the actual programming environment better, and provides a more comprehensive and detailed framework for evaluating the coding skills of LLMs.

Ranking	CodeScope (Understanding)	CodeScope (Generation)	CodeScope (Overall)	HumanEval Pass@1	MBPP Pass@1
1	WizardCoder (50.00)	GPT-4 (31.47)	GPT-4 (39.20)	GPT-4 (67.0)	GPT-4 (61.8)
2	GPT-3.5 (49.20)	GPT-3.5 (21.07)	GPT-3.5 (35.14)	WizardCoder (57.3)	Code LLaMA (57.0)
3	LLaMA 2 (48.51)	WizardCoder (9.73)	WizardCoder (29.86)	GPT-3.5 (48.1)	GPT-3.5 (52.2)
4	Code LLaMA (47.99)	Code LLaMA (8.04)	Code LLaMA (28.01)	Code LLaMA (41.5)	WizardCoder (51.8)
5	PaLM 2 (47.28)	PaLM 2 (5.46)	PaLM 2 (26.37)	PaLM 2 (37.6)	PaLM 2 (50.00)
6	GPT-4 (46.93)	StarCoder (3.86)	LLaMA 2 (25.50)	StarCoder (33.6)	LLaMA 2 (45.4)
7	Vicuna (46.54)	Vicuna (2.59)	Vicuna (24.57)	LLaMA 2 (30.5)	StarCoder (43.6)
8	StarCoder (44.04)	LLaMA 2 (2.49)	StarCoder (23.95)	Vicuna (15.2)	Vicuna (22.4)

Comparison of results of eight baseline models on CodeScope, HumanEval and MBPP benchmarks.