Novelty Assessment Report

Paper: WebDevJudge: Evaluating (M)LLMs as Critiques for Web Development Quality
PDF URL: https://openreview.net/pdf?id=CCSPm6V5EF
Venue: ICLR 2026 Conference Submission
Year: 2026
Report Generated: 2025-12-30

Abstract

The paradigm of LLM-as-a-judge is emerging as a scalable and efficient alternative to human evaluation, demonstrating strong performance on well-defined tasks. However, its reliability in open-ended tasks with dynamic environments and complex interactions remains unexplored. To bridge the gap, we introduce WebDevJudge, a systematic benchmark for assessing LLM-as-a-judge performance in web development, with support for both non-interactive evaluation based on static observations and continuous interactive evaluation with a dynamic web environment. WebDevJudge comprises human preference labels over paired web implementations, annotated with structured and query-grounded rubrics to ensure high-quality ground truth. Using this benchmark, we comprehensively evaluate various evaluators, including LLMs, MLLMs, and agentic workflows. We systematically investigate the impact of different paradigms and guidance mechanisms. Our experiments reveal a significant gap between LLM judges and human experts. In-depth analysis indicates this gap stems from fundamental model limitations, including failures in recognizing functional equivalence, verifying task feasibility, and mitigating bias. Overall, WebDevJudge presents a significant challenge to LLM-as-a-judge, offering insights to guide future research toward developing more reliable and capable automated evaluators for complicated scenarios.

Core Task Landscape

This paper addresses: **Evaluating LLM-as-a-Judge Performance in Web Development Quality Assessment**

A total of **25 papers** were analyzed and organized into a taxonomy with **8 categories**.

Taxonomy Overview

The research landscape has been organized into the following main categories:

- **LLM-as-a-Judge Evaluation Frameworks and Benchmarks**
- **LLM-as-a-Judge Reliability and Bias Analysis**
- **LLM-as-a-Judge Applications in Software Engineering**
- **LLM-Based Code Generation and Web Development Tools**

Complete Taxonomy Tree

- Evaluating LLM-as-a-Judge Performance in Web Development Quality Assessment Survey Taxonomy
- LLM-as-a-Judge Evaluation Frameworks and Benchmarks
  - Web Development and Interactive Artifact Evaluation ★ (4 papers)
  - [0] WebDevJudge: Evaluating (M)LLMs as Critiques for Web Development Quality (Anon et al., 2026) View paper
  - [2] ArtifactsBench: Bridging the Visual-Interactive Gap in LLM Code Generation Evaluation (Zhang Chenchen, 2025) View paper
  - [19] You Don't Know Until You Click: Automated GUI Testing for Production-Ready Software Evaluation (Lin Xian-hao, 2025) View paper
  - [21] IWR-Bench: Can LVLMs reconstruct interactive webpage from a user interaction video? (Chen Yang, 2025) View paper
  - Algorithmic Code Evaluation Benchmarks (3 papers)
  - [7] Codejudge: Evaluating code generation with large language models (Weiwei Tong, 2024) View paper
  - [8] Codejudgebench: Benchmarking llm-as-a-judge for coding tasks (Jiang Hongchao, 2025) View paper
  - [15] AXIOM: Benchmarking LLM-as-a-Judge for Code via Rule-Based Perturbation and Multisource Quality Calibration (Ruiqi Wang, 2025) View paper
- LLM-as-a-Judge Reliability and Bias Analysis
  - Position and Presentation Bias (2 papers)
  - [5] Judging the judges: A systematic study of position bias in llm-as-a-judge (Shi Lin, 2024) View paper
  - [18] Don't Judge Code by Its Cover: Exploring Biases in LLM Judges for Code Evaluation (Moon Ji-Won, 2025) View paper
  - General Reliability and Validation Enhancement (1 papers)
  - [11] Can External Validation Tools Improve Annotation Quality for LLM-as-a-Judge? (Arduin Findeis, 2025) View paper
- LLM-as-a-Judge Applications in Software Engineering
  - Code Review and Quality Assessment (3 papers)
  - [10] Core: Resolving code quality issues using llms (Jui Pradhan, 2024) View paper
  - [17] Benchmarking LLMs for Fine-Grained Code Review with Enriched Context in Practice (Ruida Hu, 2025) View paper
  - [20] LLMs as Code Review Agents: A Rapid Review and Experimental Evaluation with Human Expert Judges (Marcin Kawalerowicz, 2025) View paper
  - Agentic and Interactive Code Evaluation (3 papers)
  - [9] CodeVisionary: An Agent-based Framework for Evaluating Large Language Models in Code Generation (Wang Xin-Chen, 2025) View paper
  - [12] ReLook: Vision-Grounded RL with a Multimodal LLM Critic for Agentic Web Coding (Li Yuhang, 2025) View paper
  - [24] Agent-as-a-Judge: Evaluate Agents with Agents (Zhuge, 2024) View paper

    - General Software Engineering Evaluation Surveys (5 papers)
    - [3] An LLM-as-judge metric for bridging the gap with human evaluation in SE tasks (Zhou Xin, 2025) View paper
    - [4] From code to courtroom: Llms as the new software judges (He, 2025) View paper
    - [22] LLM-as-a-Judge for Software Engineering: Literature Review, Vision, and the Road Ahead (He, 2025) View paper
    - [23] Human-AI-Collaboration-For-Coding (Ravi, 2025) View paper
    - [25] Can LLMs Replace Human Evaluators? An Empirical Study of LLM-as-a-Judge in Software Engineering (Ruiqi Wang, 2025) View paper
- LLM-Based Code Generation and Web Development Tools (5 papers)
    - [1] Comparative analysis of chatbots using large language models for web development tasks (Pavel Smutny, 2024) View paper
    - [6] Conversational AI for Accessible Website Design: Integrating LLM Assistants in Website Builders (S. K. Oswal, 2025) View paper
    - [13] LLM-Based Web Generation Quality Assessment (Yizhen Gong, 2025) View paper
    - [14] Natural Language Outlines for Code: Literate Programming in the LLM Era (Kensen Shi, 2024) View paper
    - [16] Suitability of large language models for making PDF-documents more accessible and barrier-free in enterprise content management/Author Jack Heseltine (Heseltine, 2025) View paper

Narrative

Core task: evaluating LLM-as-a-judge performance in web development quality assessment. The field organizes around four main branches that reflect both methodological and application-oriented concerns. The first branch, LLM-as-a-Judge Evaluation Frameworks and Benchmarks, develops systematic testbeds and metrics to measure how well language models can serve as evaluators, with works like ArtifactsBench[2] and IWR-Bench[21] providing structured environments for assessing interactive artifacts and web-based outputs. The second branch, LLM-as-a-Judge Reliability and Bias Analysis, investigates the robustness and fairness of these automated judges, examining issues such as position bias and consistency across diverse evaluation scenarios. The third branch, LLM-as-a-Judge Applications in Software Engineering, explores practical deployment in code review, accessibility auditing, and quality assurance tasks, while the fourth branch, LLM-Based Code Generation and Web Development Tools, focuses on the generation side, producing the artifacts that judges must evaluate. Together, these branches capture the dual challenge of building reliable automated evaluators and applying them to increasingly complex software artifacts.

Several active lines of work highlight key trade-offs and open questions. One cluster examines the granularity and domain specificity of evaluation: some studies develop fine-grained rubrics for code correctness and style, while others like CodeJudge[7] and CodeJudgeBench[8] target broader functional assessments. Another theme concerns the interplay between automated judgment and human oversight, with works exploring when LLM judges align with expert reviewers and when they introduce systematic errors. WebDevJudge[0] sits within the Web Development and Interactive Artifact Evaluation cluster, emphasizing the challenge of assessing visual and interactive quality in web outputs—a setting where traditional code metrics fall short. Compared to neighbors like ArtifactsBench[2], which provides a general-purpose benchmark for interactive artifacts, and IWR-Bench[21], which focuses on web reasoning tasks, WebDevJudge[0] zeroes in on the specific reliability and validity of LLM judges when evaluating web development quality, bridging evaluation framework design with practical software engineering concerns.

Related Works in Same Category

The following **3 sibling papers** share the same taxonomy leaf node with the original paper:

1. ArtifactsBench: Bridging the Visual-Interactive Gap in LLM Code Generation Evaluation

Authors: Zhang Chenchen, Li Yuhang, Chenchen Zhang, Xu Can, Yuhang Li, et al. (53 authors total) | **Year/Venue**: 2025 • arXiv.org | **URL**: View paper

Abstract

The generative capabilities of Large Language Models (LLMs) are rapidly expanding from static code to dynamic, interactive visual artifacts. This progress is bottlenecked by a critical evaluation gap: established benchmarks focus on algorithmic correctness and are blind to the visual fidelity and interactive integrity that define modern user experiences. To bridge this gap, we introduce ArtifactsBench, a new benchmark and paradigm for the automated, multimodal evaluation of visual code generatio...

Relationship Analysis

Both papers belong to the Web Development and Interactive Artifact Evaluation category, focusing on benchmarking LLM judges for assessing web development quality through visual rendering and interactive behavior. They overlap in their core objective of evaluating LLM-as-a-judge performance on web artifacts using multimodal assessment (code + screenshots) and measuring agreement with human preferences. The key differences are that WebDevJudge emphasizes interactive evaluation with live web environments and agentic workflows, uses rubric trees for structured annotation, and focuses on continuous interaction capabilities, while ArtifactsBench prioritizes automated multimodal evaluation at scale through programmatic rendering and temporal screenshots, achieving higher correlation with human preferences (94.4% vs WebDev Arena) across a larger benchmark of 1,825 tasks.

2. You Don't Know Until You Click: Automated GUI Testing for Production-Ready Software Evaluation

Authors: Lin Xian-hao, Yutong Bian, Xie Yu-peng, Xianhao Lin, Liu Tianyang, et al. (30 authors total) | **Year/Venue**: 2025 | **URL**: View paper

Abstract

Large Language Models (LLMs) and code agents in software development are rapidly evolving from generating isolated code snippets to producing full-fledged software applications with graphical interfaces, interactive logic, and dynamic behaviors. However, current benchmarks fall short in evaluating such production-ready software, as they often rely on static checks or binary pass/fail scripts, failing to capture the interactive behaviors and runtime dynamics that define real-world usability - qua...

Relationship Analysis

Both papers belong to the Web Development and Interactive Artifact Evaluation category, focusing on benchmarks that assess LLM judges through dynamic interaction with web applications and GUI testing. They overlap in evaluating functional correctness, visual fidelity, and interactive behavior of web implementations using agent-based approaches. However, the original paper (WebDevJudge) emphasizes meta-evaluation of LLM-as-a-judge performance with human preference alignment using rubric trees, while the candidate paper (AppEvalPilot/RealDevWorld) focuses on automated end-to-end GUI testing for production-ready software repositories across multiple domains with an agent-based execution framework.

3. IWR-Bench: Can LVLMs reconstruct interactive webpage from a user interaction video?

Authors: Chen Yang, Liu, Minghao, Yang Chen, Shen Yu-fan, et al. (44 authors total) | **Year/Venue**: 2025 | **URL**: View paper

Abstract

The webpage-to-code task requires models to understand visual representations of webpages and generate corresponding code. However, existing benchmarks primarily focus on static screenshot-to-code tasks, thereby overlooking the dynamic interactions fundamental to real-world web applications. To address this limitation, this paper introduces IWR-Bench, a novel benchmark for evaluating the capabilities of Large Vision-Language Models (LVLMs) in interactive webpage reconstruction from video. IWR-Be...

Relationship Analysis

Both papers belong to the Web Development and Interactive Artifact Evaluation category, focusing on benchmarks that assess LLM judges' ability to evaluate web development quality through interactive and visual assessment. While WebDevJudge evaluates LLM-as-a-judge performance in assessing paired web implementations with human preference labels and supports both static and interactive evaluation paradigms, IWR-Bench focuses on evaluating LVLMs' capability to reconstruct interactive webpages from user interaction videos, using an agent-as-a-judge framework to assess functional correctness and visual fidelity of generated code rather than comparing existing implementations.

Contributions Analysis

Overall novelty summary. The paper introduces WebDevJudge, a benchmark for evaluating LLM-as-a-judge performance in web development quality assessment, including both static and interactive evaluation modes. It resides in the 'Web Development and Interactive Artifact Evaluation' leaf, which contains four papers total. This represents a relatively sparse research direction within the broader taxonomy of 25 papers across the field, suggesting that systematic evaluation of LLM judges specifically for web development tasks remains an emerging area with limited prior benchmarking efforts.

The taxonomy reveals that WebDevJudge sits at the intersection of evaluation frameworks and software engineering applications. Its sibling papers include ArtifactsBench (general interactive artifacts), IWR-Bench (web reasoning), and one other web-focused work. Neighboring leaves address algorithmic code evaluation (three papers) and various application domains like code review and agentic workflows. The scope notes clarify that this leaf specifically targets visual rendering, interactivity, and dynamic behavior assessment—distinguishing it from purely algorithmic correctness evaluation. This positioning suggests the work addresses a gap between general code evaluation and the specialized requirements of web artifact assessment.

Among the 29 candidates examined across three contributions, none were identified as clearly refuting the paper's claims. The WebDevJudge benchmark contribution examined nine candidates with zero refutable matches; the query-grounded rubric methodology examined ten candidates with zero refutations; and the WebDevJudge-Unit diagnostic dataset examined ten candidates, also with zero refutations. This pattern suggests that within the limited search scope, the specific combination of web development focus, interactive evaluation support, and structured rubric annotation appears relatively novel, though the analysis does not claim exhaustive coverage of all potentially relevant prior work.

Based on the top-29 semantic matches examined, the work appears to occupy a distinct niche within LLM-as-a-judge research. The sparse population of its taxonomy leaf and absence of direct refutations across contributions indicate potential novelty, though this assessment is constrained by the search methodology. The gap identified between LLM judges and human experts in web development evaluation may represent a substantive contribution to understanding judge reliability in visually-oriented, interactive domains.

This paper presents **3 main contributions**, each analyzed against relevant prior work:

Contribution 1: WebDevJudge benchmark for evaluating LLM-as-a-judge in web development

Description: The authors present WebDevJudge, a meta-evaluation benchmark designed to assess how well LLMs can judge web development quality. The benchmark supports both static code-based evaluation and interactive assessment within live web environments, addressing the gap in evaluating LLM judges on complex, dynamic tasks.

This contribution was assessed against **9 related papers** from the literature. Papers with potential prior art are analyzed in detail with textual evidence; others receive brief assessments.

---

1. Codejudge: Evaluating code generation with large language models

URL: View paper

Brief Assessment

CodeJudge[7] focuses on evaluating code generation quality through semantic correctness assessment, not on meta-evaluating LLM judges themselves. WebDevJudge specifically benchmarks how well LLMs can serve as judges for web development tasks with interactive environments, which is a distinct contribution.

---

2. Codejudgebench: Benchmarking llm-as-a-judge for coding tasks

URL: View paper

Brief Assessment

CodeJudgeBench[8] focuses on evaluating LLM-as-a-judge for coding tasks (code generation, code repair, unit test generation) rather than web development quality assessment. The candidate does not address the interactive web environment evaluation or web-specific quality metrics that are central to the original paper's contribution.

---

3. Interaction2Code: Benchmarking MLLM-based Interactive Webpage Code Generation from Interactive Prototyping

URL: View paper

Brief Assessment

Interaction2Code[51] focuses on benchmarking MLLM-based interactive webpage code generation from prototypes, not on evaluating LLM-as-a-judge capabilities for assessing web development quality. The candidate addresses a different problem domain (code generation evaluation) rather than judge evaluation.

---

4. Humanevalcomm: Benchmarking the communication competence of code generation for llms and llm agents

URL: View paper

Brief Assessment

HumanEvalComm[48] focuses on evaluating communication competence of LLMs in code generation tasks (asking clarifying questions about requirements), not on evaluating LLM-as-a-judge capabilities for web development quality assessment.

---

5. Web-bench: A llm code benchmark based on web standards and frameworks

URL: View paper

Brief Assessment

Web-Bench[50] focuses on evaluating LLM code generation accuracy for web development projects through end-to-end testing, not on assessing LLM-as-a-judge capabilities for evaluating web development quality or providing meta-evaluation of automated evaluators.

---

6. Automatic generation of benchmarks and reliable LLM judgment for code tasks

URL: View paper

Brief Assessment

Automatic Benchmark Generation[49] focuses on automatic generation of benchmarks for code tasks (translation, summarization, generation) using self-consistency loops in a code generation graph. It does not address web development evaluation or interactive assessment in live web environments, which are the core novelties of WebDevJudge.

7. Program synthesis with large language models

URL: View paper

Brief Assessment

Program Synthesis[47] focuses on synthesizing Python programs from natural language descriptions using large language models, not on evaluating LLM judges for web development quality assessment. The candidate addresses code generation tasks, while the original paper addresses meta-evaluation of LLM judges in interactive web development contexts.

8. A survey on evaluating large language models in code generation tasks

URL: View paper

Brief Assessment

Code Generation Survey[29] focuses on evaluating LLMs for general code generation tasks across various programming languages and metrics, not specifically on evaluating LLM-as-a-judge capabilities in web development quality assessment.

9. CodeJudge-eval: Can large language models be good judges in code understanding?

URL: View paper

Brief Assessment

CodeJudge-Eval[46] focuses on evaluating LLMs as judges for code correctness in competitive programming tasks, not web development quality assessment. The candidate addresses code generation correctness judgment, while the original paper evaluates web development quality with both static and interactive components.

Contribution 2: Query-grounded rubric tree annotation methodology

Description: The authors develop a structured annotation approach using rubric trees that break down web development requirements into hierarchical, verifiable criteria. This methodology achieves high inter-annotator agreement (89.7%) and provides reliable ground-truth preference labels for the benchmark.

This contribution was assessed against **10 related papers** from the literature. Papers with potential prior art are analyzed in detail with textual evidence; others receive brief assessments.

1. AI-Driven Automated Test Generation Framework for VCU: A Multidimensional Coupling Approach Integrating Requirements, Variables and Logic

URL: View paper

Brief Assessment

VCU Test Generation[44] focuses on automotive software testing with variable binding and executable code generation, not on rubric-based annotation methodologies for decomposing requirements into hierarchical verifiable criteria for evaluation purposes.

2. An Efficient Rubric-based Generative Verifier for Search-Augmented LLMs

URL: View paper

Brief Assessment

Rubric Generative Verifier[42] focuses on nugget-based rubrics for search-augmented LLMs in QA tasks, not on web development quality assessment with hierarchical rubric trees achieving high inter-annotator agreement.

3. ProImage-Bench: Rubric-Based Evaluation for Professional Image Generation

URL: View paper

Brief Assessment

ProImage-Bench[43] focuses on rubric-based evaluation for professional image generation tasks, not web development. The rubric structures serve different purposes: image quality assessment versus web implementation verification. The domains and evaluation targets are fundamentally different.

4. Qa-lign: Aligning llms through constitutionally decomposed qa

URL: View paper

Brief Assessment

QA-Lign[37] focuses on constitutional alignment of LLMs through decomposed Q&A programs for safety evaluation, not on web development annotation methodologies. The rubric structures serve different purposes in different domains.

5. Multidimensional Rubric-oriented Reward Model Learning via Geometric Projection Reference Constraints

URL: View paper

Brief Assessment

Multidimensional Rubric Reward[45] focuses on medical LLM alignment using rubric-based reward modeling for reinforcement learning, not on annotation methodology for creating ground-truth preference labels in web development benchmarks.

6. ExpertLongBench: Benchmarking Language Models on Expert-Level Long-Form Generation Tasks with Structured Checklists

URL: View paper

Brief Assessment

ExpertLongBench[40] focuses on expert-level task evaluation with domain-specific rubrics for long-form generation tasks, while the original paper develops rubric trees specifically for web development quality assessment with hierarchical, verifiable criteria achieving 89.7% inter-annotator agreement. The candidate's rubrics are designed for different domains and evaluation purposes.

7. A framework for specification-based testing

URL: View paper

Brief Assessment

Specification-Based Testing[38] focuses on test template frameworks for deriving tests from formal specifications (Z notation), not on annotation methodologies for decomposing requirements into hierarchical verifiable criteria for human preference labeling.

8. Aecbench: A hierarchical benchmark for knowledge evaluation of large language models in the aec field

URL: View paper

Brief Assessment

AECBench[39] focuses on rubric-based evaluation for AEC domain knowledge assessment, not web development requirements decomposition. The rubrics are used for scoring LLM-generated documents against expert criteria, rather than annotating preference labels for web implementations.

9. Spoq: Scaling {Machine-Checkable} systems verification in coq

URL: View paper

Brief Assessment

SpoQ[41] focuses on machine-checkable systems verification in Coq for system software, not on annotation methodologies for web development requirements. The paper does not address rubric-based annotation or hierarchical requirement decomposition for evaluation purposes.

10. Pencils down! automatic rubric-based evaluation of retrieve/generate systems

URL: View paper

Brief Assessment

Pencils Down[36] focuses on decomposing queries into test questions for evaluating IR/RAG system responses, not on hierarchical rubric trees for annotating web development implementations with verifiable criteria and achieving high inter-annotator agreement.

Contribution 3: WebDevJudge-Unit diagnostic dataset for feasibility verification

Description: The authors create WebDevJudge-Unit, a targeted dataset of 502 test cases designed to diagnose and evaluate the ability of LLM-based and agent-based evaluators to verify whether specific web development tasks are feasible and correctly implemented.

This contribution was assessed against **10 related papers** from the literature. Papers with potential prior art are analyzed in detail with textual evidence; others receive brief assessments.

1. mHumanEval-a multilingual benchmark to evaluate large language models for code generation

URL: View paper

Brief Assessment

mHumanEval[32] focuses on multilingual code generation benchmarks with natural language prompts in 204 languages, not on diagnostic datasets for evaluating feasibility verification capabilities of LLM-based evaluators in web development contexts.

2. Codescore: Evaluating code generation by learning code execution

URL: View paper

Brief Assessment

CodeScore[28] focuses on evaluating code generation through functional correctness and execution metrics, not on creating diagnostic datasets for feasibility verification in web development tasks.

3. An empirical study of the non-determinism of chatgpt in code generation

URL: View paper

Brief Assessment

ChatGPT Non-Determinism[26] focuses on non-determinism in code generation across multiple runs, not on creating diagnostic datasets for evaluating feasibility verification capabilities of LLM-based evaluators in web development contexts.

4. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation

URL: View paper

Brief Assessment

ChatGPT Rigorous Evaluation[31] focuses on augmenting test cases for code generation benchmarks to evaluate functional correctness, not on creating diagnostic datasets for feasibility verification in web development tasks or agent-based evaluators.

5. Evaluating the Test Adequacy of Benchmarks for LLMs on Code Generation

URL: View paper

Brief Assessment

Test Adequacy Benchmarks[33] focuses on evaluating test case quality in code generation benchmarks (HumanEval, MBPP) using coverage and mutation metrics, not on creating diagnostic datasets for feasibility verification in web development tasks.

6. Beyond correctness: Benchmarking multi-dimensional code generation for large language models

URL: View paper

Brief Assessment

Multi-Dimensional Code Generation[35] focuses on evaluating code generation across readability, maintainability, correctness, and efficiency dimensions for general programming tasks. It does not address feasibility verification in web development contexts or create diagnostic datasets for evaluating LLM-based judges' ability to verify task feasibility in interactive web environments.

7. A survey on evaluating large language models in code generation tasks

URL: View paper

Brief Assessment

Code Generation Survey[29] discusses evaluation metrics for code generation but does not present a diagnostic dataset specifically designed to evaluate feasibility verification capabilities of LLM-based or agent-based evaluators in web development contexts.

8. Evaluating large language models in class-level code generation

URL: View paper

Brief Assessment

Class-Level Code Generation[27] focuses on evaluating LLMs on class-level Python code generation tasks, not on diagnostic datasets for feasibility verification in web development or implementation correctness evaluation.

9. GeoJSEval: an automated evaluation framework for large language models on JavaScript-based geospatial computation and visualization code generation

URL: View paper

Brief Assessment

GeoJSEval[30] focuses on evaluating LLMs for JavaScript-based geospatial code generation with function-level test cases, not on diagnostic datasets for evaluating LLM-based or agent-based evaluators' feasibility verification capabilities in web development tasks.

10. Execution-based evaluation for data science code generation models

URL: View paper

Brief Assessment

Data Science Execution[34] focuses on execution-based evaluation for data science code generation in Jupyter notebooks, not on diagnostic datasets for feasibility verification in web development tasks. The domains and evaluation objectives are fundamentally different.

Appendix: Text Similarity Detection

No high-similarity text segments were detected across any compared papers.

References

- [0] WebDevJudge: Evaluating (M)LLMs as Critiques for Web Development Quality View paper
- [1] Comparative analysis of chatbots using large language models for web development tasks View paper
- [2] ArtifactsBench: Bridging the Visual-Interactive Gap in LLM Code Generation Evaluation View paper
- [3] An LLM-as-judge metric for bridging the gap with human evaluation in SE tasks View paper
- [4] From code to courtroom: Llms as the new software judges View paper
- [5] Judging the judges: A systematic study of position bias in llm-as-a-judge View paper
- [6] Conversational AI for Accessible Website Design: Integrating LLM Assistants in Website Builders View paper
- [7] Codejudge: Evaluating code generation with large language models View paper
- [8] Codejudgebench: Benchmarking llm-as-a-judge for coding tasks View paper
- [9] CodeVisionary: An Agent-based Framework for Evaluating Large Language Models in Code Generation View paper
- [10] Core: Resolving code quality issues using llms View paper
- [11] Can External Validation Tools Improve Annotation Quality for LLM-as-a-Judge? View paper
- [12] ReLook: Vision-Grounded RL with a Multimodal LLM Critic for Agentic Web Coding View paper
- [13] LLM-Based Web Generation Quality Assessment View paper
- [14] Natural Language Outlines for Code: Literate Programming in the LLM Era View paper
- [15] AXIOM: Benchmarking LLM-as-a-Judge for Code via Rule-Based Perturbation and Multisource Quality Calibration View paper
- [16] Suitability of large language models for making PDF-documents more accessible and barrier-free in enterprise content management/Author Jack Heseltine View paper
- [17] Benchmarking LLMs for Fine-Grained Code Review with Enriched Context in Practice View paper
- [18] Don't Judge Code by Its Cover: Exploring Biases in LLM Judges for Code Evaluation View paper
- [19] You Don't Know Until You Click: Automated GUI Testing for Production-Ready Software Evaluation View paper
- [20] LLMs as Code Review Agents: A Rapid Review and Experimental Evaluation with Human Expert Judges View paper
- [21] IWR-Bench: Can LVLMs reconstruct interactive webpage from a user interaction video? View paper
- [22] LLM-as-a-Judge for Software Engineering: Literature Review, Vision, and the Road Ahead View paper
- [23] Human-AI-Collaboration-For-Coding View paper
- [24] Agent-as-a-Judge: Evaluate Agents with Agents View paper
- [25] Can LLMs Replace Human Evaluators? An Empirical Study of LLM-as-a-Judge in Software Engineering View paper
- [26] An empirical study of the non-determinism of chatgpt in code generation View paper
- [27] Evaluating large language models in class-level code generation View paper
- [28] Codescore: Evaluating code generation by learning code execution View paper
- [29] A survey on evaluating large language models in code generation tasks View paper
- [30] GeoJSEval: an automated evaluation framework for large language models on JavaScript-based geospatial computation and visualization code generation View paper
- [31] Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation View paper
- [32] mHumanEval-a multilingual benchmark to evaluate large language models for code generation View paper
- [33] Evaluating the Test Adequacy of Benchmarks for LLMs on Code Generation View paper
- [34] Execution-based evaluation for data science code generation models View paper
- [35] Beyond correctness: Benchmarking multi-dimensional code generation for large language models View paper
- [36] Pencils down! automatic rubric-based evaluation of retrieve/generate systems View paper
- [37] Qa-lign: Aligning llms through constitutionally decomposed qa View paper
- [38] A framework for specification-based testing View paper
- [39] Aecbench: A hierarchical benchmark for knowledge evaluation of large language models in the aec field View paper
- [40] ExpertLongBench: Benchmarking Language Models on Expert-Level Long-Form Generation Tasks with Structured Checklists View paper
- [41] Spoq: Scaling {Machine-Checkable} systems verification in coq View paper
- [42] An Efficient Rubric-based Generative Verifier for Search-Augmented LLMs View paper
- [43] ProImage-Bench: Rubric-Based Evaluation for Professional Image Generation View paper
- [44] AI-Driven Automated Test Generation Framework for VCU: A Multidimensional Coupling Approach Integrating Requirements, Variables and Logic View paper
- [45] Multidimensional Rubric-oriented Reward Model Learning via Geometric Projection Reference Constraints View paper
- [46] CodeJudge-eval: Can large language models be good judges in code understanding? View paper
- [47] Program synthesis with large language models View paper
- [48] Humanevalcomm: Benchmarking the communication competence of code generation for llms and llm agents View paper
- [49] Automatic generation of benchmarks and reliable LLM judgment for code tasks View paper
- [50] Web-bench: A llm code benchmark based on web standards and frameworks View paper

- [51] Interaction2Code: Benchmarking MLLM-based Interactive Webpage Code Generation from Interactive Prototyping View paper