EDA Bench: An Execution-Based Benchmark for Language Model Agents on Functional PCB Construction

Anonymous Author(s)
Affiliation
Address
email

Abstract

Language-model agents are increasingly able to operate electronic design automation tools: they can search datasheets, select components, edit KiCad projects, run design checks, and iterate after failures. Existing evaluations, however, rarely test the property that matters most for printed-circuit-board reconstruction: whether the produced board preserves the external electrical behavior of the target design. We introduce EDA Bench, an execution-based benchmark for agentic KiCad PCB reconstruction. EDA Bench contains 35 released tasks derived from public open-hardware projects spanning connector breakouts, power-path circuits, microcontroller systems, motor drivers, camera and compute adapters, and high-density carriers. Each harness receives a prompt and a standard KiCad environment, then must produce a complete KiCad project. Submissions are graded by an oracle that parses the realized PCB, extracts routed copper, checks declared external ports, models trace parasitics and short faults, runs ngspice simulations, and applies KiCad ERC and DRC checks. The score is based on observable external-I/O behavior rather than text overlap, screenshot similarity, component inventory, or reference-designator matching. We validate the artifact with reference, fail, and mutation canaries: all 35 released frozen references score 1.0, and all 35 released structural-failure canaries score at most 0.15. Initial baselines show that EDA Bench is not saturated by current evaluated harnesses. The strongest evaluated web-enabled harness scores 4.58% overall, building 21 of 35 projects while failing most functional checks. These results identify a substantial gap between producing syntactically valid or visually plausible KiCad artifacts and reconstructing boards that behave correctly at their electrical interfaces.

1 Introduction

Language-model agents can now operate development tools over long horizons. In software, this has made execution-based evaluation a central benchmark design pattern: an agent modifies an artifact, runs tools, observes failures, and is evaluated by executable checks rather than by surface-form similarity. SWE-bench evaluates agents on real GitHub issues whose solutions must pass repository tests [Jimenez et al., 2023]. OSWorld extends execution-based evaluation to open-ended desktop and web tasks in real computer environments [Xie et al., 2024]. ProgramBench evaluates a still broader reconstruction setting in which agents rebuild software from an executable and documentation, with behavioral tests used to compare against the reference program [Yang et al., 2026]. These benchmarks share a common principle: the evaluated artifact should be judged by behavior in an execution environment.

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

Hardware design has an analogous workflow. Engineers search for parts, inspect datasheets, draft schematics, route boards, run ERC and DRC checks, simulate where possible, and produce manufacturing files. For electronic design automation, however, execution-based testing is substantially harder. Correctness depends on routed connectivity, electrical behavior, manufacturability constraints, and external interfaces rather than only on executable program behavior. Existing CAD and EDA benchmarks cover important parts of this space, including text-to-CAD generation [Khan et al., 2024], geometric CAD validation [Epoch AI, 2026, CAD Arena, 2026, Barkley et al., 2026], schematic generation [Al Hasan et al., 2026, Zou et al., 2026, Luo et al., 2026], schematic understanding [Lu et al., 2026], board-level schematic design [Qiu et al., 2026], hardware bug repair [Cui et al., 2026], and PCB placement and routing reasoning [Li et al., 2026]. These evaluations motivate execution-based assessment for engineering artifacts, but they do not directly test whether an agent can reconstruct a routed KiCad PCB that preserves the target board's external electrical behavior.

EDA Bench addresses this gap by evaluating complete agent harnesses on source-backed KiCad PCB reconstruction tasks. A harness receives a fixed task prompt and runtime assets, then must write a complete KiCad project in `/workspace/final_project`. KiCad provides an open-source EDA environment for schematic capture, integrated simulation, PCB layout, 3D rendering, and manufacturing export [KiCad Project, 2026]. The released tasks are based on public hardware projects, and the default evaluated harnesses are web-enabled. This creates a tradeoff between reproducibility and realism similar to public software-engineering benchmarks: public sources make the benchmark auditable, but web-enabled agents may benefit from locating and adapting upstream material.

The benchmark is built around a functional scoring principle: evaluate the observable behavior of the realized board at its external electrical interfaces. A submitted project is parsed, checked, simulated, and compared against task contracts. ngspice supplies the open-source circuit-simulation backend used by the oracle [ngspice contributors, 2026]. A board can score well with a different internal implementation if it exposes the required external behavior. It can also fail with a visually plausible layout if external nets are missing, shorted, swapped, isolated, or not driven by the required circuitry. Although this evaluation requires more engineering effort than syntax checks, visual matching, or component-inventory comparison, positive scores are more informative because the submitted board must satisfy observable electrical-interface checks.

Contributions. This paper makes five contributions.

- We introduce EDA Bench, a public benchmark of 35 source-backed KiCad PCB reconstruction tasks spanning connector breakouts, power-path circuits, controller boards, compute adapters, motor drivers, and high-density carrier boards.
- We define a reproducible harness protocol in which agents receive only task-visible runtime packs and must produce complete KiCad projects, while references, source snapshots, contracts, canaries, and oracle internals remain part of the grading pack.
- We develop an execution-based PCB scoring oracle that evaluates realized board behavior through external-I/O contracts, routed-copper extraction, ngspice simulation, task-family checks, active-circuit realization checks, KiCad ERC and DRC diagnostics, and manufacturability caps.
- We validate the scoring artifact with frozen references, structural fail canaries, mutation canaries, and invariance controls, showing that the oracle accepts human-designed references while rejecting common function-breaking PCB failures.
- We report public-provenance baselines for six web-enabled harness configurations, including final projects, grader outputs, usage summaries, cost estimates, and regrade commands, and show that current evaluated frontier-model harnesses remain far from functional PCB reconstruction.

2 Benchmark overview

EDA Bench evaluates PCB reconstruction as an execution task. For each task, a harness receives a fixed prompt, runtime task pack, tool environment, model configuration, and access policy, then must produce a complete KiCad project in `/workspace/final_project`. The submitted project

Table 1: EDA Bench evaluation pipeline. Each submission is evaluated as a complete harness output rather than as text, a screenshot, or a component list.

Stage	Artifact or operation
Task input	The harness receives a fixed prompt, runtime task pack, tool environment, model configuration, and access policy.
Harness execution	The agent operates in the Dockerized KiCad environment and writes a project to `/workspace/final_project`.
Project parsing	The grader locates KiCad files and extracts connectors, pads, nets, board outline, and routed copper geometry.
Contract binding	External ports from the submitted board are matched to task-declared I/O contracts and expected measurements.
Execution checks	The grader applies KiCad ERC and DRC diagnostics, ngspice simulations, routed-I/O checks, parasitic models, and short-fault checks.
Score reporting	The oracle returns a normalized task score, hard-cap diagnostics, build status, and provenance artifacts.

is graded automatically. The oracle parses the KiCad files, extracts external connectors and routed copper, applies task-declared stimuli and probes, runs simulation where applicable, checks ERC and DRC diagnostics, and assigns a normalized score based on the realized board behavior.

Table 1 summarizes the execution and grading pipeline. The benchmark is source-backed. Each released task is derived from a public hardware project and has a frozen reference design, but the evaluated runtime pack does not mount the reference project, source snapshot, canaries, grader contracts, or oracle implementation. This split makes the artifact auditable while preserving a clean separation between agent-visible inputs and grader-only materials. Because the released tasks are public-source and the default baselines are web-enabled, EDA Bench measures PCB reconstruction under a declared public-source access policy rather than closed-book circuit invention.

Scores should be interpreted as properties of complete evaluated harnesses, not isolated model weights. A harness may include prompting, web search, source discovery, KiCad tool use, repair loops, and project-generation logic. This systems-level framing is intentional because practical EDA automation depends on whether an agent can operate the full workflow, not only on whether a model can emit plausible schematic or layout text in one shot.

The central scoring claim is functional rather than visual. A board can score well with a different internal implementation if it exposes the required external electrical behavior under the task contract. Conversely, a board can fail despite looking plausible if external nets are missing, isolated, shorted, swapped, unrouted, or not driven by the required active circuitry. The score is therefore not a fabrication certificate, regulatory review, thermal analysis, EMC validation, or vendor-accurate silicon simulation. It is an execution-based measure of declared external-I/O behavior under the released oracle.

3 Benchmark tasks

EDA Bench contains 35 released source-backed tasks. Each task is anchored to a public upstream hardware project and a frozen reference design. The public task set covers six difficulty levels, shown in Table 2. Five additional candidate tasks were quarantined from the release because their upstream redistribution terms require permission or acquisition-script replacement.

The task count reflects the cost of producing score-supported hardware evaluation instances rather than a preference for small benchmarks. Each released task requires a source design, a frozen KiCad reference, runtime and grading pack separation, an external-I/O contract, canary submissions, license triage, and oracle support. We therefore prioritize tasks with executable functional checks over a larger collection of weakly specified or weakly checked prompts.

The task catalog includes compact breakouts and converters such as `usb_c_female_breakout`, `mcp23017_breakout`, and `rs485_transceiver_breakout`. It also includes power and control boards such as `bq24295_power_path_board`, `picopd`, and `robotont_driver_board`, compute and camera or display adapters such as `cm4_csi_adapter`, `m2_pcie_adapter`,

Table 2: Task distribution and representative categories in EDA Bench.

Difficulty	Tasks	Representative task families
Very easy	2	Compact connector breakouts and small adapters
Easy	5	I/O breakouts, logic-level conversion, USB/UART basics
Medium	4	Power-path boards, serial transceivers, small controller boards
Hard	12	Compute adapters, sensor/camera boards, multi-connector carriers
Very hard	7	Dense mixed-signal boards and larger interface carriers
Extreme	5	Large baseboards and high-density compute boards

and `ov9281_dual_camera_board`, and large carrier boards such as `cm4_baseboard`, `jetson_orin_baseboard`, and `zynq_som`.

Each task has two artifact forms. The runtime task pack is mounted for the evaluated agent and contains the prompt, task metadata, and runtime assets. It does not include the reference project, source snapshot, canaries, grader contracts, or oracle implementation. The full grading pack is public and contains the reference project, upstream snapshot, canary submissions, license notes, `io_contract.json`, `functional_contract.json`, and local tests. This split supports reproducibility while avoiding direct mounting of the answer into the evaluated container.

4 Harness protocol

A harness is evaluated as a complete system. It receives the task prompt and must produce a KiCad project in `/workspace/final_project`. Built-in harnesses run inside Docker with KiCad, ngspice, Python, Node, internet access, and the configured model interface. The current evaluated harnesses are Codex CLI with GPT-5.5 web search at low, medium, high, and xhigh reasoning, Pi with Gemini 3.1 Pro Preview web high, and Pi with DeepSeek V4 Pro web high.

This protocol evaluates system architecture, not only a model weight. A one-shot generator, a tool-using agent, and a multi-agent harness can all target the same task interface. This distinction matters because EDA work is iterative: the model may need to create files, run KiCad tools, inspect failures, search for upstream sources, and repair invalid output.

Each run uploads provenance to Hugging Face. Reports include prompts, command lines, usage summaries, grader outputs, final projects, and generated artifacts. Large Docker image uploads and raw model transcripts can be disabled to minimize storage usage, but reports still retain byte-count summaries for omitted transcript content.

5 Scoring oracle

The scorer evaluates a submitted KiCad project through explicit I/O simulation. Unsupported, missing, or malformed submissions receive zero score. Supported tasks declare their external ports, stimuli, probes, simulator models, expected measurements, and tolerances. For each submission, the oracle performs seven steps. It locates and parses the submitted KiCad project, runs KiCad ERC and DRC diagnostics, extracts connectors, pads, nets, and routed copper geometry, binds submitted external ports to the task contract, constructs task-declared simulation probes and stimuli, computes task-family measurements and hard caps, and returns a normalized score with diagnostic artifacts.

The primary score is a normalized task score in $[0, 1]$. A benchmark aggregate is a per-task difficulty-weighted mean:

$$S = \frac{\sum_{i=1}^{N} w(d_i) s_i}{\sum_{i=1}^{N} w(d_i)}, \quad w(d_i) \in \{1.0, 1.5, 2.0, 3.0, 4.0, 5.0\}. \tag{1}$$

We also report the unweighted mean over tasks. Build success means that the submission produced a KiCad project that the grader could parse and evaluate. It is not equivalent to functional success.

The main score comes from three observable properties of the submitted PCB: external connector semantics, routed-copper behavior, and task-required active and manufacturing plausibility. The oracle parses the project, extracts PCB geometry, applies deterministic ngspice stimuli at declared

Table 3: Representative task-family checks layered on the generic external-I/O oracle. These are calibrated bindings on measured oracle signals, not vendor-accurate silicon or channel models.

Task family	Declared behavior	Common hard caps
Connector breakouts and simple adapters	Connector pins and power/ground pads, continuity, shorts, routed coverage, pin/net semantic agreement	Swapped pins, split nets, isolated pads, power-ground shorts
Serial and level-shifting interfaces	USB/UART/RS-485/logic-side boundary ports, signal transfer families, active-device realization	Missing active path, weak external mapping, missing routed I/O
Power-path and charger boards	Input, battery/load, configuration, and status ports, power path presence, external safety	Missing active power role, shorts, missing component-function profile
Camera/display/compute carriers	High-density connector families and differential pairs, routed families, pair plausibility	Missing high-speed pair families, weak routing, shorts, split nets
Large mixed-signal baseboards	Multi-rail power, GPIO, debug, serial, high-speed, and peripheral interfaces	Incomplete connector semantics, missing active realization, shorts

external ports, samples boundary waveforms, and applies hard caps for failures such as missing routed copper, swapped external pins, shorts, missing active circuitry, unpowered active devices, invalid high-speed pair families, and gross fabrication implausibility. Exact reference designators, component inventory, footprint similarity, outline similarity, and route-shape similarity are diagnostics, not primary score signals.

Within each task, declared measurements produce normalized sub-scores for external-port mapping, routed continuity, simulation behavior, and task-family realization checks. These sub-scores are combined according to the task contract to form a soft functional score. Hard caps then impose upper bounds for structural failures that make the board externally incorrect or physically implausible, such as shorts, missing routed I/O, swapped connector pins, or absent active circuitry. The final task score is the resulting capped functional score. This design allows partial credit for partially correct external behavior while preventing submissions with severe structural faults from receiving high scores through unrelated checks.

6 Artifact validation

Because EDA Bench uses a task-specific oracle, the release includes controls that calibrate score interpretation. The pass canary is the frozen reference project and must score 1.0 under the same oracle. The fail canary removes active internal circuitry when available. Passive boards fall back to external boundary removal. The fail canary must score below the release threshold, currently 0.75.

Oracle validity. We validate the oracle as a contract-level evaluator, not as a vendor-accurate electrical simulator. Its intended validity criterion is whether score changes track engineering-relevant changes to declared external-I/O behavior. Positive controls test that human-designed reference boards are realizable and score 1.0. Negative controls test that structural changes a PCB engineer would regard as function-breaking receive low scores. Invariance controls test that benign representation changes, such as reference-designator renaming or plausible route-width edits, do not dominate the score.

A full local sweep on May 6, 2026 verified all 35 released task canaries: every pass canary scored 1.0, and every fail canary scored at most 0.15. The generated-mutation sweep graded 162 mutation canaries across the original 40-task candidate set. All tasks passed, with every reference scoring 1.0 and every structural fail canary scoring at most 0.15. Mutations covered missing board outlines, unrouted external nets, isolated external pads, power or signal pads tied to ground, and missing high-speed-like routes for high-speed carrier tasks.

Table 4: Oracle-validity evidence. The controls calibrate the released score as an external-I/O contract evaluator rather than as a complete manufacturing or vendor-silicon review.

Evidence	What it supports
35/35 released frozen human-designed reference PCBs score 1.0	The tasks are solvable, and the oracle accepts real upstream designs under their declared contracts.
35/35 released structural fail canaries score at most 0.15	The score is not reducible to file existence, KiCad syntax, or component-shell matching.
162 candidate-set mutation canaries pass validation	The oracle is sensitive to shorts, unrouted nets, isolated pads, missing outlines, and missing high-speed-like routing.
Refdes-renamed USB-C board scores 1.0	The score does not depend on exact connector reference designators.
Plausible USB-C route-width edit remains at least 0.85	The score does not require exact route-shape or width matching when behavior remains plausible.
Tiny-trace USB-C control scores at most 0.35	Manufacturability caps affect scores for physically implausible routing.

Table 5: Complete web-enabled harness baselines. The primary score is difficulty-weighted, and the unweighted score is the mean task score.

Harness	Weighted	Unweighted	Builds
Codex GPT-5.5 web low	0.0124	0.0143	31/35
Codex GPT-5.5 web medium	0.0279	0.0314	35/35
Codex GPT-5.5 web high	0.0293	0.0229	11/35
Codex GPT-5.5 web xhigh	0.0458	0.0500	21/35
Pi Gemini 3.1 Pro Preview web high	0.0113	0.0086	1/35
Pi DeepSeek V4 Pro web high	0.0000	0.0000	0/35

These controls do not show that the oracle replaces expert board review or captures vendor-specific silicon behavior. They show that, for the benchmark's declared external-I/O contracts, the score accepts human-designed reference boards, rejects common function-breaking PCB mutations, and is invariant to several superficial implementation details.

7 Experiments

We evaluate six web-enabled harness configurations on the 35 released tasks. All runs use the same task prompts, task packs, Docker image, and grading protocol. Table 5 reports a May 6, 2026 no-model regrade of saved final projects under the current oracle revision.

No evaluated harness achieves a high absolute score. The main empirical finding is that build success substantially overestimates functional PCB reconstruction. The medium reasoning-level Codex harness builds all 35 released projects but scores only 2.79%. The xhigh harness builds fewer projects than medium but scores higher. These results indicate that executable KiCad output is not a sufficient metric for EDA agents and that reasoning and tool-use choices affect the distribution of functional failures.

Difficulty breakdowns further indicate that the benchmark is not saturated by easy tasks. The xhigh harness reaches 13.75% on medium tasks and 7.08% on hard tasks, but only 2.00% on extreme tasks. The gap grows as boards require larger external contracts, active circuitry, dense routing, or high-speed interfaces.

Statistical interpretation. The current result table reports complete released-task sweeps, not repeated independent sampling campaigns. We therefore do not claim statistically significant pairwise model rankings. The claims supported by the table are qualitative and artifact-level: the benchmark is not saturated by current evaluated harnesses, build success is not functional success, and access policy materially affects the evaluated harnesses under the same task and grader revision.

Table 6: Completed no-web calibration runs. These rows are reported separately from the default web leaderboard because access policy is an evaluated condition.

Harness	Weighted	Unweighted	Builds
Codex GPT-5.5 no-web medium	0.0000	0.0000	0/35
Codex GPT-5.5 no-web high	0.0000	0.0000	0/35
Codex GPT-5.5 no-web xhigh	0.0000	0.0000	0/35
Codex GPT-5.4 no-web xhigh	0.0000	0.0000	0/35
Codex GPT-5.4-mini no-web medium	0.0000	0.0000	0/35
Codex GPT-5.4-mini no-web high	0.0000	0.0000	0/35

7.1 Access-policy calibration

EDA Bench reports access policy as part of the evaluated system. Six no-web Codex calibration sweeps use the same released tasks and grader revision as the web-enabled runs, but disable web access. All six scored zero and produced no buildable projects, indicating that the default prompts and built-in Codex harnesses rely on public-source discovery as well as KiCad operation. These rows are calibration runs rather than closed-book capability measurements for all possible harnesses, so we report them separately from the web-enabled leaderboard.

8 Failure analysis

The results support three benchmark-level conclusions. First, current evaluated frontier-model harnesses are far from solving functional PCB reconstruction: the best evaluated configuration scores only 4.58% overall on the released task set. Second, syntactic validity is not sufficient: a harness can produce buildable KiCad projects for every released task and still fail most functional checks. Third, harness architecture matters: Codex GPT-5.5 configurations with different reasoning settings produce different build and score tradeoffs, so EDA Bench evaluates complete systems rather than base models alone.

Saved grading artifacts expose the main failure modes. Among buildable Codex GPT-5.5 xhigh submissions, the most common caps were split required external-net continuity, isolated external-I/O pads, wrong external-I/O pin/net mapping, missing or weak component-function realization, missing or implausible high-speed pair geometry, and weak active-device power integrity. The all-building Codex medium run shows a similar pattern: wrong external mapping, missing component-function realization, split required external nets, same-layer trace shorts, and isolated external-I/O pads.

These caps correspond to concrete functional failures rather than parser failures. Submissions often instantiate connector footprints with plausible labels, but leave required I/O pads isolated, split required nets across disconnected copper, swap external pin semantics, omit active circuitry, or route high-speed-like interfaces implausibly. These errors explain why KiCad parseability and component presence are weak proxies for functional PCB reconstruction.

9 Related work

Execution-based software-agent benchmarks. Execution-based software benchmarks evaluate agents by running the artifacts they produce. SWE-bench frames software engineering as repository-level issue resolution, where an agent edits a real codebase and must pass tests associated with the target issue [Jimenez et al., 2023]. OSWorld evaluates multimodal agents in real desktop and web environments with custom execution-based evaluation scripts [Xie et al., 2024]. ProgramBench studies a more reconstructive setting in which agents rebuild software from an executable and documentation, and behavioral tests are generated to compare the candidate program with the reference executable [Yang et al., 2026]. EDA Bench adopts this behavioral-evaluation perspective but applies it to PCB reconstruction, where correctness depends on realized geometry, external connectivity, and electrical-interface behavior rather than software tests alone.

CAD generation and geometric evaluation. Text2CAD introduced a large-scale text-to-parametric-CAD dataset and model for generating CAD command sequences from natural-language prompts

[Khan et al., 2024]. CadEval and CAD Arena provide evaluation settings for text-to-CAD models using geometric validity, rendering, and prompt-level comparisons [Epoch AI, 2026, CAD Arena, 2026]. CADSmith further emphasizes programmatic geometric validation for generated CadQuery models [Barkley et al., 2026]. These works motivate execution-based evaluation for engineering artifacts, but their primary correctness signals are geometric rather than electrical. PCB reconstruction requires connectivity, routing, short avoidance, active-circuit realization, and external-I/O behavior in addition to shape or layout plausibility.

Circuit, schematic, and PCB benchmarks. Recent EDA benchmarks study several adjacent tasks. CircuitLM and PCBSchemaGen generate or synthesize circuit schematics from natural-language prompts and use structural or constraint-based checks to improve schematic validity [Al Hasan et al., 2026, Zou et al., 2026]. SchGen studies PCB schematic generation using semantic-grounded code representations [Luo et al., 2026]. OmniSch evaluates multimodal schematic understanding and schematic-to-graph reasoning over real-world diagrams [Lu et al., 2026]. HWE-Bench evaluates board-level schematic design with datasheet context and hardware bug repair over repository-scale hardware projects [Qiu et al., 2026, Cui et al., 2026]. PCB-Bench evaluates LLM reasoning over placement, routing, and PCB design comprehension using text, images, and complete PCB projects [Li et al., 2026]. EDA Bench differs by requiring an agent to produce a complete routed KiCad PCB project and by scoring the realized board through external-I/O simulation and PCB-geometry checks rather than schematic-only validity, visual reasoning, or placement/routing question answering.

10 Limitations

EDA Bench is not a hidden-answer or closed-book benchmark. The released tasks are public-source, and upstream designs may be discovered through web search. This determines the supported claim: scores measure EDA task solving under fixed prompts, tools, task-pack revisions, provenance, and access policy, not closed-book invention.

The task set is intentionally small compared with text benchmarks. It has 35 released tasks because each task requires a real source design, a reference project, task contracts, canaries, license review, and grader support. Breadth should grow through additional source-backed packs or a sealed private split, not by adding weakly checked or specified tasks.

The oracle is functional but not a substitute for manufacturing review. It uses deterministic generic behavioral models. It does not prove electrical safety, regulatory compliance, high-speed protocol compliance, thermal reliability, power sequencing correctness, or production readiness. Richer per-task SPICE, IBIS, S-parameter, and field-solver models would improve fidelity. The current calibrated task-family checks are deliberately conservative: they tighten measured oracle signals for four representative board families, but they are not vendor-accurate silicon or channel simulations.

Task artifacts inherit mixed upstream hardware licenses. Five candidate tasks with unlicensed, upstream-terms-only, noncommercial, or otherwise restricted source snapshots are quarantined from the public release until they receive explicit redistribution permission or acquisition-script replacements.

11 Conclusion

EDA Bench evaluates whether language-model agents can reconstruct PCB designs that function at their electrical interfaces. It is built from real public hardware projects, scored by explicit I/O simulation, and evaluated as a full harness rather than a single text response. Initial results show that current systems often produce syntactically valid KiCad artifacts, but rarely produce boards that satisfy functional boundary contracts. This discrepancy is the central evaluation target of EDA Bench.

References

Khandakar Shakib Al Hasan, Syed Rifat Raiyan, Hasin Mahtab Alvee, and Wahid Sadik. CircuitLM: A Multi-Agent LLM-Aided Design Framework for Generating Circuit Schematics from Natural Language Prompts. arXiv:2601.04505, 2026. URL: https://arxiv.org/abs/2601.04505.

Jesse Barkley, Rumi Loghmani, and Amir Barati Farimani. CADSmith: Multi-Agent CAD Generation with Programmatic Geometric Validation. arXiv:2603.26512, 2026. URL: https://arxiv.org/abs/2603.26512.

CAD Arena. CAD Arena: Open Benchmark for AI-Generated Parametric CAD. 2026. URL: https://cadarena.dev/.

Fan Cui, Hongyuan Hou, Zizhang Luo, Chenyun Yin, and Yun Liang. HWE-Bench: Benchmarking LLM Agents on Real-World Hardware Bug Repair Tasks. arXiv:2604.14709, 2026. URL: https://arxiv.org/abs/2604.14709.

Epoch AI. CadEval. 2026. URL: https://epoch.ai/benchmarks/cad-eval.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? arXiv:2310.06770, 2023. URL: https://arxiv.org/abs/2310.06770.

Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin Sheikh, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. Text2CAD: Generating Sequential CAD Models from Beginner-to-Expert Level Text Prompts. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL: https://arxiv.org/abs/2409.17106.

KiCad Project. KiCad Documentation. 2026. URL: https://docs.kicad.org/.

Jindong Li, Lianrong Chen, Bin Yang, Jiadong Zhu, Ying Wang, Yuzhe Ma, and Menglin Yang. PCB-Bench: Benchmarking LLMs for Printed Circuit Board Placement and Routing. In *International Conference on Learning Representations*, 2026. URL: https://openreview.net/forum?id=Q5QLu7XTWx.

Taiting Lu, Kaiyuan Lin, Yuxin Tian, Yubo Wang, Muchuan Wang, Sharique Khatri, Akshit Kartik, Yixi Wang, Amey Santosh Rane, Yida Wang, Yifan Yang, Yi-Chao Chen, Yincheng Jin, and Mahanth Gowda. OmniSch: A Multimodal PCB Schematic Benchmark For Structured Diagram Visual Reasoning. arXiv:2604.00270, 2026. URL: https://arxiv.org/abs/2604.00270.

Qinpei Luo, Ruichun Ma, Xinyu Zhang, and Lili Qiu. SchGen: PCB Schematic Generation with Semantic-Grounded Code Representations. OpenReview preprint, 2026. URL: https://openreview.net/forum?id=TyWs6rWWHb.

ngspice contributors. ngspice, the open-source SPICE circuit simulator. 2026. URL: https://ngspice.sourceforge.io/.

Weibo Qiu, Yinhao Xiao, and Runyu Pan. HWE-Bench: Can Language Models Perform Board-level Schematic Designs? arXiv:2603.18102, 2026. URL: https://arxiv.org/abs/2603.18102.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. arXiv:2404.07972, 2024. URL: https://arxiv.org/abs/2404.07972.

John Yang, Kilian Lieret, Jeffrey Ma, Parth Thakkar, Dmitrii Pedchenko, Sten Sootla, Emily McMilin, Pengcheng Yin, Rui Hou, Gabriel Synnaeve, Diyi Yang, and Ofir Press. ProgramBench: Can Language Models Rebuild Programs From Scratch? arXiv:2605.03546, 2026. URL: https://arxiv.org/abs/2605.03546.

Huanghaohe Zou, Peng Han, Emad Nazerian, and Alex Q. Huang. PCBSchemaGen: Constraint-Guided Schematic Design via LLM for Printed Circuit Boards. arXiv:2602.00510, 2026. URL: https://arxiv.org/abs/2602.00510.

A Artifact availability and provenance

The source repository contains the runner and harness code, task-pack loader and grader support, public documentation, package tests, and the task-pack manifest. Python wheels and source distributions exclude concrete task instances, reference projects, upstream snapshots, and canary submissions. The Hugging Face task dataset contains runtime task packs, full grader task packs, prompts, references, upstream snapshots, canary submissions, task metadata, Croissant metadata, and license/provenance notes.

The current public task dataset is hosted at `https://huggingface.co/datasets/eda-bench-neurips-2026/eda-bench-tasks`. The paper results use task-pack revision `c64b9169d6d8c0d73614e14c7050cf036aeb1559`. The public provenance dataset is hosted at `https://huggingface.co/datasets/eda-bench-neurips-2026/eda-bench-provenance`. The public code release is hosted at `https://github.com/CAD-bench/eda-bench`.

Run provenance includes task prompts, command lines, usage summaries, grader outputs, final projects, and generated artifacts. Storage-light campaigns can omit full Docker image uploads and raw model transcripts, but retain final reports, final projects, grading artifacts, and byte-count summaries for omitted transcript content.

B Validation controls

Table 7: Reviewer-facing validation controls for score interpretation.

Control	Observed result
Frozen reference project	35/35 released references score 1.0
Structural fail canary	35/35 released fail canaries score at most 0.15, below the 0.75 release threshold
Generated mutation canaries	162 generated mutation canaries across the original 40-task candidate set pass validation
Calibrated task-family checks	USB-C, RS-485, BQ24295, and M.2/PCIe references score 1.0 and their mutation canaries remain below threshold
Refdes-invariant equivalent board	USB-C reference with renamed connector refdes scores 1.0
Missing routed copper	USB-C and NanoUPDI route-removal controls score 0.0
Plausible route-width edit	USB-C route-width edit remains at least 0.85
Implausibly tiny traces	Manufacturability cap applies and USB-C tiny-trace control scores at most 0.35
Missing or unparsable project	Returns 0.0 and build failure

Two additional generic mutation families are implemented experimentally for future validation: swapped external connector pins and signal-bearing active devices with power/ground pads disconnected. These should be staged in release packs or reported as validated only after a full canary sweep.

C Failure-mode diagnostics

D Baseline provenance

The May 6, 2026 result table regrades saved final projects without rerunning model harnesses. Source runs were: `codex__gpt-5.5-web-low_20260505T200208Z_2719115`, `codex__gpt-5.5-web-medium_20260506T010715Z_2812175`, `codex__gpt-5.5-web-medium_20260506T010715Z_2812182`, `codex__gpt-5.5-web-medium_20260506T032737Z_2884928`, `codex__gpt-5.5-web-medium_20260506T010716Z_2812190`, `codex__gpt-5.5-web-high_20260505T200208Z_2719113`, `codex__gpt-5.5-web-xhigh_20260506T032737Z_2884940`, `codex__gpt-5.5-web-xhigh_20260506T032737Z_2884938`, `codex__gpt-5.5-web-xhigh_20260506T032737Z_2884943`, `codex__gpt-5.5-web-xhigh_20260506T034126Z_2891808`,

Table 8: Common functional failure modes exposed by the oracle. These failures are typical of buildable submissions and illustrate why KiCad parseability is not sufficient for functional PCB reconstruction.

Failure mode	Observable symptom	Why it affects score
Split required external nets	Pins that should belong to the same external interface are assigned to disconnected copper regions or separate nets	Boundary stimuli cannot propagate through the declared interface, even when connector footprints are present
Isolated external I/O pads	Required pads are placed but not connected to routed copper or active circuitry	The board exposes a physical pad without realizing the electrical behavior required by the contract
Wrong pin or net mapping	External connector pins are swapped, mislabeled, or attached to semantically incorrect nets	The board may appear visually plausible while driving or observing the wrong external signal
Missing active realization	Components required for level shifting, power-path behavior, transceiver behavior, or control logic are absent or not connected	Passive connectivity alone cannot implement the task-required circuit behavior
Weak active-device power integrity	Active devices are present but lack required power, ground, or enable connectivity	The simulated or inferred behavior cannot be attributed to a realizable powered circuit
Implausible high-speed geometry	Differential or high-speed-like interface families are missing, weakly routed, or routed with implausible geometry	Dense carrier and compute tasks require not only pin labels but routed interface families with plausible physical realization

`pi__gemini-3.1-pro-preview-web-high_20260505T184850Z_2697190`, and `pi__deepseek__deepseek-v4-pro-web-high_20260505T200208Z_2719116`.

The executable regrade entry point is `uv run regrade-provenance`. For example, the Codex medium row is reproduced by passing its four source runs:

```
uv run regrade-provenance \
  --run codex__gpt-5.5-web-medium_20260506T010715Z_2812175 \
  --run codex__gpt-5.5-web-medium_20260506T010715Z_2812182 \
  --run codex__gpt-5.5-web-medium_20260506T032737Z_2884928 \
  --run codex__gpt-5.5-web-medium_20260506T010716Z_2812190
```

By default the command uses the current released task manifest, downloads saved `submission/final_project` directories from Hugging Face provenance, and runs the current local oracle.

Completed no-web calibration runs are present under these provenance paths: `evals/harnesses/codex__gpt-5.5-no_web-medium_20260507T033319Z_3432012`, `evals/harnesses/codex__gpt-5.5-no_web-high_20260507T033319Z_3432018`, `evals/harnesses/codex__gpt-5.5-no_web-xhigh_20260507T033319Z_3432017`, `evals/harnesses/codex__gpt-5.4-no_web-xhigh_20260507T033528Z_3439753`, `evals/harnesses/codex__gpt-5.4-mini-no_web-medium_20260507T033724Z_3448333`, and `evals/harnesses/codex__gpt-5.4-mini-no_web-high_20260507T033724Z_3448341`.

E Cost and compute accounting

The completed GPT-5.5 web runs used storage-light provenance. The combined Codex GPT-5.5 low/medium/high/xhigh campaign uploaded about 185.6 MiB of Hugging Face provenance. Across all six final harnesses, the total uploaded provenance was about 193.1 MiB.

The estimates are computed from recorded token usage, current provider pricing, and the Codex token-based rate card. They are included to make leaderboard submissions auditable and to discourage hidden cost and provenance blowups.

Table 9: Recorded Codex GPT-5.5 web campaign accounting.

Harness	Estimated OpenAI API USD	Estimated Codex credits	HF provenance
Codex low	38.22	955.41	48.9 MB
Codex medium	84.40	2110.05	49.5 MB
Codex high	49.39	1234.64	27.6 MB
Codex xhigh	182.75	4568.64	68.6 MB

F Release and license gate

EDA Bench separates runtime prompts from full task packs with references, snapshots, canaries, and grader contracts. For NeurIPS release, full packs with unlicensed, upstream-terms-only, noncommercial, or otherwise restricted source snapshots must be handled by one of three policies: obtain explicit redistribution permission, remove the source snapshot and provide a reproducible acquisition script, or quarantine the task from the public full pack while retaining runtime metadata.

The current public release quarantines five candidate tasks under this conservative rule: `cm5io_official`, `mixed_signal_stm32_dev_board`, `rp2350b_dev_board`, `stm32f_audio_codec`, and `ef28_badge`. The public score-supported full-pack release has 35 tasks. These task ids may remain documented as candidates, but their references, source snapshots, and canaries are not part of the released full-pack artifact.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's scope as introducing an execution-based benchmark for web-enabled PCB reconstruction agents, documenting its harness protocol and external-I/O oracle, validating the scoring artifact with canaries, and reporting initial harness baselines. The paper explicitly excludes hidden-answer circuit invention, fabrication readiness, regulatory certification, and vendor-accurate electrical validation.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Limitations section discusses public-source task exposure, task-set size, oracle fidelity, and upstream license constraints. The benchmark overview and access-policy calibration sections also clarify that the default baselines are web-enabled and should not be interpreted as closed-book PCB invention.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.

- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs.
- The authors should reflect on the factors that influence the performance of the approach.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that are not acknowledged in the paper.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is a benchmark and empirical evaluation paper and does not include theoretical results or proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper reports the task-pack revision, harness protocol, scoring formula, runtime and grading artifact split, provenance dataset, source run identifiers, and regrade procedure. The code, released task artifacts, grading artifacts, and saved final projects are public or described through the release and quarantine policy.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Appendix A identifies the source repository, Hugging Face task dataset, task-pack revision, Croissant metadata, and public provenance dataset. Appendix F describes the license triage policy and states that five upstream-license-risk candidate tasks are quarantined from the released full-pack artifact.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer.
- The instructions should contain the exact command and environment needed to run to reproduce the results.
- The authors should provide instructions on data access and preparation.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines.
- At submission time, to preserve anonymity, the authors should release anonymized versions, if applicable.
- Providing as much information as possible in supplemental material is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The benchmark overview, harness protocol, scoring oracle, artifact validation, and experiments sections describe the task interface, Dockerized tool environment, model and harness configurations, score definition, build-success definition, task count, validation controls, and access-policy separation.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The paper states that the reported results are complete released-task sweeps rather than repeated independent sampling campaigns. It therefore does not claim statistically significant pairwise model rankings. The supported conclusions are artifact-level and qualitative: the benchmark is not saturated by current evaluated harnesses, build success diverges from functional score, and access policy affects the evaluated harnesses under the reported protocol.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated.
- The method for calculating the error bars should be explained.
- The assumptions made should be given.
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- For asymmetric distributions, the authors should be careful not to show symmetric error bars that would yield results that are out of range.
- If error bars are reported in tables or plots, the authors should explain how they were calculated and reference the corresponding figures or tables.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper specifies the Dockerized KiCad, ngspice, Python, and Node environment, records run identifiers and provenance, and reports cost and provenance-size accounting for the main Codex campaign in Appendix E. Saved run provenance includes additional execution details for reproduced runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is a benchmark and evaluation study without human subjects or deception. The paper discusses limitations, release constraints, and the risk of overtrust in generated hardware artifacts.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper frames the benchmark as a tool for measuring EDA-agent reliability and explicitly states that scores are not evidence of manufacturing readiness, safety, EMC compliance, regulatory compliance, or production readiness. The main negative impact is overtrust in generated hardware artifacts. The mitigation is conservative scoring, validation controls, and explicit scope limits.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses, fairness considerations, privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, when it is being used as intended but gives incorrect results, and from misuse.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases benchmark tasks, grading assets, and evaluation code rather than a new high-risk generative model or scraped personal or sensitive dataset. Upstream-license-risk hardware snapshots are quarantined from the released full-pack artifact until permission or acquisition-script replacement is available.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model.
    - Datasets that have been scraped from the Internet could pose safety risks.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Appendix F states the license triage policy and identifies the five quarantined candidate tasks. The repository includes `LICENSES.md`, task-pack license notes, and source provenance metadata.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license should be included for each asset.
    - For scraped data from a particular source, the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper documents the task taxonomy, runtime and full-pack split, scoring oracle, validation controls, provenance structure, Croissant metadata, artifact availability, and release policy. The repository and dataset contain detailed task and release documentation.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets if applicable.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human-subjects research.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval or equivalent may be required for any human subjects research.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity, if applicable, such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not important, original, or non-standard components of the core method development in this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.