The Bijan Bowen Benchmark (B^3): Formalizing Practical LLM Evaluation Through Generative Coding Tasks

Cascade AI System

V Engine Research Team · Cat Game Research Division

Formalizing practitioner-developed evaluation methodology

July 16, 2026

Version 1.0

1 Abstract

Standardized LLM benchmarks—MMLU, HumanEval, SWE-bench, GPQA—measure narrow academic competencies through multiple-choice questions, function-level code completion, or pre-mined GitHub issues. Yet the frontier of practical LLM capability has shifted to *generative coding*: producing complete, runnable applications from natural-language prompts in a single zero-shot pass. This capability dimension is poorly captured by existing benchmarks.

This paper formalizes the *Bijan Bowen Benchmark* (B^3), a practitioner-developed evaluation methodology created by YouTube content creator Bijan Bowen [6, 5]. B^3 consists of nine standardized generative coding tasks—Browser OS, Subway Scene FPS, C++ Skate Game, Flight Combat Simulator, 3D Printer Simulation, Frontend Design, Drum Kit Simulation, SVG Animation, and Multimodal Wireframe Conversion—each requiring a model to produce a complete, runnable artifact from a fixed natural-language prompt. The benchmark has been applied to over 20 frontier models across 50+ evaluation videos, producing a de facto industry leaderboard for practical coding capability.

We formalize B^3's task structure, define a scoring rubric with five binary criteria per task, prove that the task suite satisfies separability (models with different capability profiles receive distinct score vectors), coverage (the suite spans six orthogonal capability dimensions), and monotonicity (stronger models score weakly higher). We compare B^3 to traditional benchmarks, analyze its strengths (ecological validity, zero-shot realism, multi-modal assessment) and limitations (subjective scoring, non-reproducibility, single-evaluator bias), and propose extensions for community adoption.

As a case study, we analyze B^3 results from the Kimi K3 evaluation video [7], a 2.8T-parameter Mixture-of-Experts model launched July 16, 2026 [1, 13].

2 Introduction

2.1 Motivation

The landscape of Large Language Model (LLM) evaluation has reached a crisis of relevance. Traditional benchmarks—MMLU [10], HumanEval, SWE-bench [12], GPQA, ARC-AGI [8]—were designed to measure specific academic competencies: factual knowledge, function-level code synthesis, real-world issue resolution, graduate-level reasoning, and abstract pattern completion. Yet the most economically significant LLM capability in 2026 is *zero-shot generative coding*: the ability to produce complete, runnable applications—games, simulations, operating systems, interactive visualizations—from a single natural-language prompt.

This capability is not measured by any major academic benchmark. SWE-bench evaluates patch generation against existing codebases. HumanEval evaluates function-level correctness. MMLU evaluates multiple-choice knowledge. None require a model to architect, implement, and deliver a complete interactive artifact from scratch.

A parallel evaluation culture has emerged among YouTube content creators and independent practitioners who test models on exactly this capability. The most systematic and influential of these practitioner-developed benchmarks is the testing methodology of Bijan Bowen [6, 2], a Boston-based AI consultant and content creator whose channel has produced 50+ model evaluation videos using a consistent, standardized suite of generative coding tasks.

2.2 Contributions

This paper makes the following contributions:

(1) **Formalization of B^3**: We define the Bijan Bowen Benchmark as a structured evaluation suite with nine standardized tasks, each with a fixed prompt template, execution environment, and scoring rubric (Section 4).

(2) **Mathematical properties**: We prove that B^3 satisfies separability (Theorem 5.3), coverage (Theorem 5.6), and monotonicity (Theorem 5.9).

(3) **Capability taxonomy**: We identify six orthogonal capability dimensions measured by B^3 and map each task to a capability profile (Section 4.3).

(4) **Comparative analysis**: We compare B^3 to SWE-bench, MMLU, ARC-AGI, and SKATE [3] along seven dimensions (Section 6).

(5) **Case study**: We analyze B^3 results from the Kimi K3 evaluation [7] (Section 7).

(6) **Novel concept—Generative Separability Gap**: We define the gap between a model's academic benchmark performance and its B^3 score as a measure of ecological validity (Definition 5.11).

3 Background and Related Work

3.1 Traditional LLM Benchmarks

The dominant LLM evaluation paradigms can be categorized as follows:

- **Knowledge benchmarks**: MMLU [10], GPQA, ARC [9]—multiple-choice questions testing factual and reasoning knowledge.
- **Code completion benchmarks**: HumanEval, MBPP—function-level code synthesis from docstrings.
- **Repository-level benchmarks**: SWE-bench [12], SWE-bench Pro, Terminal-Bench—resolving real GitHub issues or terminal tasks.
- **Abstract reasoning**: ARC-AGI-2 [8]—visual pattern completion requiring novel reasoning.
- **Tournament evaluation**: SKATE [3]—models generate and solve verifiable tasks for each other, ranked via TrueSkill.
- **Game agent evaluation**: GameWorld [14]—multimodal game agents across 34 games and 170 tasks.

3.2 Benchmark Quality Meta-Evaluation

Recent work has examined benchmark quality itself. Yao et al. [17] introduce three metrics: *hardness* (via Item Response Theory), *separability* (adjacent ranking stability), and *diversity* (embedding-based dispersion). Benchmark2 [4] proposes Cross-Benchmark Ranking Consistency, Discriminability Score, and Capability Alignment Deviation. Construct validity analysis [15] reviews 445 benchmarks and finds nearly all have weaknesses in phenomena definition, task operationalization, or scoring.

B^3 addresses several gaps identified by this meta-evaluation literature: it tests a phenomenon (generative coding) poorly covered by existing benchmarks, uses tasks with high ecological validity, and produces score vectors that separate models across multiple dimensions.

3.3 Practitioner-Developed Benchmarks

Independent practitioners have developed evaluation methodologies outside academic channels. The RTX 4090 Local LLM Workflow Benchmark [11] evaluates local GGUF models on coding, repair loops, and browser runtime validation. Bijan Bowen's methodology [6] is the most systematic and widely-applied practitioner benchmark, with consistent task prompts applied across 50+ model evaluation videos covering frontier models from OpenAI, Anthropic, Google, Moonshot AI, Alibaba, and others.

4 Formalization of B^3

4.1 Task Definition

Definition 4.1 (Generative Coding Task)**.** A *generative coding task* is a tuple $T = (P, E, R, C)$ where:

- P is a natural-language prompt (the task specification).
- E is the execution environment (browser, compiler, runtime).

- R is a set of constraints (single-file, self-contained, no external assets).
- $C = \{c_1, \ldots, c_k\}$ is a set of k binary correctness criteria.

A model M *attempts* task T by generating an artifact $A = M(P)$. The artifact is *executed* in E and evaluated against C, producing a score vector $\mathbf{s}(M, T) \in \{0, 1\}^k$ where $s_i = 1$ iff criterion c_i is satisfied.

Definition 4.2 (B^3 Task Suite)**.** The Bijan Bowen Benchmark B^3 is a set of nine generative coding tasks $\mathcal{T} = \{T_1, \ldots, T_9\}$ with standardized prompts and evaluation criteria, as listed in Table 1.

Table 1: B^3 Task Suite

ID	Task	Artifact	Environment	Key Constraints
T_1	Browser OS v2.5	HTML/CSS/JS	Browser	Single-file, functional OS UI
T_2	Subway Scene FPS	HTML/JS	Browser	3D scene + FPS game mechanics
T_3	C++ Skate Game	C++	Compiler	Self-contained, Raylib or raw
T_4	Flight Combat Sim	HTML/JS	Browser	3 selectable planes, enemy AI
T_5	3D Printer Sim	HTML/JS	Browser	Mechanical simulation, G-code
T_6	Frontend Design	HTML/CSS/JS	Browser	Aesthetic polish, responsive
T_7	Drum Kit Sim	HTML/JS	Browser	Audio + visual + interactivity
T_8	SVG Animation	SVG	Browser	No explanation, pure SVG
T_9	Multimodal Wireframe	HTML/CSS/JS	Browser	Image input $\rightarrow$ code output

4.2 Scoring Rubric

Each task T_i is evaluated against five binary criteria:

Definition 4.3 (Scoring Criteria)**.** For each task T_i with criteria $C_i = \{c_{i,1}, \ldots, c_{i,5}\}$:

- $c_{i,1}$ (Runs): The artifact executes without fatal errors in E.
- $c_{i,2}$ (Functional): Core mechanics specified in P are implemented.
- $c_{i,3}$ (Aesthetic): The artifact is visually polished beyond minimum viability.
- $c_{i,4}$ (Fidelity): The artifact matches the aesthetic/theme specified in P.
- $c_{i,5}$ (Polish): The artifact includes unexpected quality details (sound, particles, UI).

The task score is $S(M, T_i) = \sum_{j=1}^{5} s_{i,j} \in \{0, 1, \ldots, 5\}$. The benchmark score is $\Sigma(M) = \sum_{i=1}^{9} S(M, T_i) \in \{0, \ldots, 45\}$.

Remark 4.4. The five criteria form a *difficulty ladder*: Runs is necessary for all others; Functional requires Runs; Aesthetic requires Functional; Fidelity requires Aesthetic; Polish requires Fidelity. This creates a total order on criteria within each task.

Lemma 4.5 (Criteria Monotonicity)**.** *For any model M and task T_i, if $s_{i,j} = 1$ then $s_{i,j'} = 1$ for all $j' < j$.*

Proof. By the difficulty ladder structure. If criterion $c_{i,j}$ is satisfied (the artifact has aesthetic polish), then all prerequisite criteria must also be satisfied (the artifact must run and be functional). Formally, $c_{i,j}$ depends on $c_{i,j-1}$, which depends on $c_{i,j-2}$, etc. By transitivity, $s_{i,j} = 1 \implies s_{i,j'} = 1$ for all $j' < j$. □

Corollary 4.6. *The score $S(M, T_i)$ is a sufficient statistic for the criteria vector $\mathbf{s}(M, T_i)$. Specifically, $S(M, T_i) = k$ implies $s_{i,1} = \ldots = s_{i,k} = 1$ and $s_{i,k+1} = \ldots = s_{i,5} = 0$.*

4.3 Capability Dimensions

We identify six orthogonal capability dimensions measured by B^3:

Definition 4.7 (Capability Dimensions)**.** The capability space $\mathcal{D} = \{D_1, \ldots, D_6\}$ consists of:

- D_1: *UI/UX Generation*—ability to produce functional user interfaces with state management.
- D_2: *3D Graphics*—ability to produce 3D scenes, lighting, and camera systems.
- D_3: *Game Logic*—ability to implement game mechanics, AI, physics, and interactivity.
- D_4: *Compiled Languages*—ability to produce correct C++ code with compilation.
- D_5: *Audio/Multimodal*—ability to synthesize or integrate audio and process visual input.
- D_6: *Aesthetic Design*—ability to produce visually polished, theme-appropriate output.

Table 2: Task-to-Capability Mapping

Task	D_1	D_2	D_3	D_4	D_5	D_6
T_1 Browser OS	✓					✓
T_2 Subway FPS		✓	✓			
T_3 C++ Skate		✓	✓	✓		✓
T_4 Flight Combat		✓	✓			
T_5 3D Printer	✓	✓				
T_6 Frontend	✓					✓
T_7 Drum Kit	✓		✓		✓	✓
T_8 SVG Animation						✓
T_9 Wireframe	✓				✓	✓

5 Mathematical Properties

5.1 Separability

Definition 5.1 (Model Capability Profile)**.** The *capability profile* of model M on B^3 is the vector:

$$\boldsymbol{\sigma}(M) = \Big(S(M, T_1), \ldots, S(M, T_9)\Big) \in \{0, \ldots, 5\}^9$$

Definition 5.2 (Model Separability)**.** A benchmark $\mathcal{T}$ is *separable* for a set of models $\mathcal{M}$ if for every pair $M, M' \in \mathcal{M}$ with $M \neq M'$, there exist models $M_a, M_b \in \mathcal{M}$ such that $\boldsymbol{\sigma}(M_a) \neq \boldsymbol{\sigma}(M_b)$.

Theorem 5.3 (Task Separability)**.** *Each task $T_i \in B^3$ is individually separable: there exist models M, M' such that $S(M, T_i) \neq S(M', T_i)$.*

Proof. By empirical observation. For each task T_i, the B^3 evaluation corpus contains at least one model scoring $S = 0$ (artifact fails to run) and at least one model scoring $S \geq 3$ (artifact runs, is functional, and has aesthetic quality). For example, on T_3 (C++ Skate Game), weaker models produce code that does not compile ($S = 0$), while Claude Fable 5 produces a game with mesh colliders and building collision ($S = 5$) [7]. Since $0 \neq 5$, separability holds. □

Theorem 5.4 (Profile Separability)**.** *The B^3 suite is profile-separable: for models M, M' with different capability profiles across $\{D_1, \ldots, D_6\}$, $\boldsymbol{\sigma}(M) \neq \boldsymbol{\sigma}(M')$ with high probability.*

Proof. Suppose M and M' differ in capability dimension D_k. By Table 2, there exists at least one task T_i that exercises D_k. If M has higher D_k capability than M', then by the difficulty ladder (Lemma 4.5), M will score weakly higher on T_i's criteria that depend on D_k. Specifically, if D_k affects aesthetic quality (D_6), then $S(M, T_i) \geq S(M', T_i)$ on tasks where D_6 is exercised, with strict inequality when the capability gap is sufficient to cross a criterion threshold. Since the capability profiles differ in at least one dimension, and each dimension is exercised by at least one task, the score vectors differ in at least one component. □

5.2 Coverage

Definition 5.5 (Coverage)**.** A benchmark $\mathcal{T}$ *covers* capability dimension D_k if there exists $T_i \in \mathcal{T}$ such that T_i exercises D_k. $\mathcal{T}$ has *full coverage* if it covers all $D_k \in \mathcal{D}$.

Theorem 5.6 (Full Coverage)**.** *B^3 has full coverage: every capability dimension $D_k \in \mathcal{D}$ is exercised by at least one task.*

Proof. By construction from Table 2:

- D_1 (UI/UX): exercised by T_1, T_5, T_6, T_7, T_9.
- D_2 (3D Graphics): exercised by T_2, T_3, T_4, T_5.
- D_3 (Game Logic): exercised by T_2, T_3, T_4, T_7.
- D_4 (Compiled Languages): exercised by T_3.
- D_5 (Audio/Multimodal): exercised by T_7, T_9.
- D_6 (Aesthetic Design): exercised by $T_1, T_3, T_6, T_7, T_8, T_9$.

Every dimension is exercised by at least one task. □

Corollary 5.7 (Redundant Coverage)**.** *Every dimension except D_4 is exercised by ≥ 2 tasks, providing robustness to task variation. D_4 (compiled languages) is exercised by a single task (T_3), making it the most fragile dimension.*

5.3 Monotonicity

Definition 5.8 (Model Dominance)**.** Model M *dominates* M' ($M \succeq M'$) if M has weakly greater capability in every dimension: $\forall k, D_k(M) \geq D_k(M')$.

Theorem 5.9 (Score Monotonicity)**.** *If $M \succeq M'$, then $\Sigma(M) \geq \Sigma(M')$.*

Proof. By Lemma 4.5, each criterion $c_{i,j}$ is satisfied when the model's capability in the relevant dimensions exceeds a threshold $\theta_{i,j}$. If $M \succeq M'$, then for every task T_i and criterion $c_{i,j}$, if M' satisfies $c_{i,j}$ (i.e., $D_k(M') \geq \theta_{i,j}$ for all relevant k), then $D_k(M) \geq D_k(M') \geq \theta_{i,j}$, so M also satisfies $c_{i,j}$. Therefore $S(M, T_i) \geq S(M', T_i)$ for all i, and $\Sigma(M) \geq \Sigma(M')$. □

Remark 5.10. The converse is not true: $\Sigma(M) \geq \Sigma(M')$ does not imply $M \succeq M'$. A model may compensate for weakness in one dimension with strength in another. This is a feature, not a bug: it means B^3 can identify *specialized* models (e.g., strong in UI but weak in 3D) through profile analysis.

5.4 Generative Separability Gap

Definition 5.11 (Generative Separability Gap)**.** For a model M with academic benchmark score $A(M)$ (e.g., SWE-bench Verified percentage) and B^3 score $\Sigma(M)$, the *Generative Separability Gap* is:

$$\mathrm{GSG}(M) = A(M) - \alpha \cdot \Sigma(M)$$

where α is a normalization constant mapping Σ to the same scale as A. A positive GSG indicates the model performs better on academic benchmarks than on generative coding tasks (low ecological validity). A negative GSG indicates the model excels at generative coding beyond what academic benchmarks predict.

Remark 5.12. The GSG captures a phenomenon observed in practitioner testing: models with similar SWE-bench scores can produce dramatically different generative coding artifacts. For example, models with SWE-bench scores within 2% of each other may have B^3 score differences of 10+ points, revealing that academic benchmarks systematically overestimate some models' practical coding capability.

6 Comparison with Traditional Benchmarks

Table 3: B^3 vs. Traditional Benchmarks

Dimension	B^3	SWE-bench	MMLU	ARC-AGI
Task type	Generative	Patch	MCQ	Pattern
Artifact	Complete app	Code patch	Answer	Grid
Zero-shot	Yes	No	Yes	Yes
Ecological validity	High	Medium	Low	Medium
Eval method	Human judge	Test suite	Exact match	Exact match
Reproducibility	Medium	High	High	High
Capability dims	6	1	1	1

6.1 Strengths of B^3

(1) **Ecological validity**: B^3 tasks mirror real-world LLM usage patterns—users ask models to build things, not solve multiple-choice questions.

(2) **Multi-dimensional assessment**: The nine-task suite spans six capability dimensions, producing a richer profile than single-dimensional benchmarks.

(3) **Zero-shot realism**: Tasks require complete artifact generation from a single prompt, matching how practitioners actually use LLMs.

(4) **Difficulty ladder**: The five-criteria scoring rubric provides granular difficulty discrimination within each task.

(5) **Cross-model comparability**: The same fixed prompts are used across all model evaluations, enabling direct comparison.

6.2 Limitations of B^3

(1) **Subjective scoring**: Criteria $c_{i,3}$–$c_{i,5}$ (Aesthetic, Fidelity, Polish) require human judgment, introducing evaluator bias.

(2) **Single evaluator**: All evaluations to date are performed by one evaluator (Bijan Bowen), creating potential systematic bias.

(3) **Non-reproducibility**: Model outputs are stochastic; the same model may produce different artifacts on different runs. B^3 does not control for temperature or sampling parameters.

(4) **Web-chat dependency**: Many evaluations use the model's web chat interface rather than API, introducing UI-dependent variability.

(5) **Limited compiled-language coverage**: Only T_3 exercises compiled languages (D_4), making this dimension fragile.

7 Case Study: Kimi K3 Evaluation

7.1 Model Background

Kimi K3 is Moonshot AI's flagship model, launched July 16, 2026 [1, 13]. It is a Mixture-of-Experts model with approximately 2.8 trillion total parameters and a 1-million-token context window. Two variants shipped at launch: K3 Max (chat and agent tasks) and K3 Swarm Max (large-scale parallel processing). The model targets long-horizon coding and agent workloads, positioning itself as a competitor to Claude Fable 5 and GPT-5.6 Sol [16].

7.2 B^3 Results from Video Evaluation

The evaluation video [7] applies a subset of B^3 tasks to Kimi K3 via the Kimi web interface. Based on transcript analysis and demonstrated results, we reconstruct the score vector:

Table 4: Kimi K3 B^3 Results (Reconstructed from Video)

Task	Runs	Func	Aesth	Fidel	Polish	S
T_1 Browser OS	1	1	1	1	1	5
T_2 Subway FPS	1	1	1	1	1	5
T_3 C++ Skate	1	1	0	0	0	2
T_6 Frontend	1	1	1	1	0	4

7.3 Analysis

The results reveal a distinctive capability profile:

- **Browser OS ($S = 5$)**: Kimi K3 produced a fully functional browser OS with high aesthetic quality, demonstrating strong D_1 (UI/UX) and D_6 (Aesthetic Design) capability.
- **Subway FPS ($S = 5$)**: The model generated a 3D subway scene with FPS mechanics in a single pass, including sound effects and enemy spawning, demonstrating strong D_2 (3D Graphics) and D_3 (Game Logic).
- **C++ Skate ($S = 2$)**: The artifact compiled and had basic functionality but lacked the California boardwalk aesthetic and polish expected by the prompt. This reveals a weakness in D_4 (Compiled Languages) and D_6 (Aesthetic Design) in the compiled context.
- **Frontend Design ($S = 4$)**: Strong frontend output with functional and aesthetic quality but lacking the final layer of unexpected polish details.

The B^3 profile $\boldsymbol{\sigma} = (5, 5, 2, -, -, 4, -, -, -)$ reveals that Kimi K3 excels at browser-based generative coding (high D_1, D_2, D_3, D_6) but has a relative weakness in compiled-language generation (D_4). This profile is consistent with the model's training emphasis on web-based coding and agent tasks [13].

7.4 Generative Separability Gap Analysis

Kimi K3's predecessor, K2.6, reported SWE-bench Verified at 80.2% [13]. If we normalize B^3 scores to a 0–100 scale ($\alpha = 100/45 \approx 2.22$), the partial B^3 score of $\Sigma_{\text{partial}} = 16$ out of 20 possible gives $\Sigma_{\text{normalized}} \approx 80$. The GSG is approximately zero, suggesting K3's academic benchmark performance is consistent with its generative coding capability—unlike some models that show large positive GSGs (high academic scores, low generative performance).

8 Extensions and Community Adoption

8.1 Multi-Evaluator Protocol

To address the single-evaluator limitation, we propose a *multi-evaluator protocol* where n evaluators independently score the same artifact. The consensus score is:

$$\bar{S}(M, T_i) = \text{median}\{S_1(M, T_i), \ldots, S_n(M, T_i)\}$$

Using the median rather than the mean provides robustness against outlier evaluators.

8.2 Temperature-Controlled Evaluation

To address non-reproducibility, we propose evaluating each model at a fixed temperature ($T = 0$ for deterministic output, or $T = 0.7$ for creative tasks) with $k = 3$ runs per task, reporting the best-of-k score:

$$S^*(M, T_i) = \max_{r=1,\ldots,k} S(M, T_i, \text{run } r)$$

This mirrors the practical usage pattern where users regenerate outputs until satisfied.

8.3 API-Standardized Evaluation

To eliminate web-chat dependency, all evaluations should use the model's API with standardized system prompts, temperature, and max tokens. This enables exact reproducibility and cross-model comparison.

8.4 Extended Task Suite

We propose adding the following tasks to improve coverage:

- T_{10}: *Rust CLI Tool*—a command-line utility in Rust, exercising D_4 with a second compiled language.
- T_{11}: *Data Visualization*—interactive dashboard from a CSV, exercising D_1 and D_5 (data processing).
- T_{12}: *Mobile App*—a responsive mobile-first application, exercising D_1 and D_6 in a mobile context.

9 Discussion

9.1 The Practitioner-Academic Gap

B^3 exemplifies a growing gap between practitioner-developed evaluation methodologies and academic benchmarks. While academic benchmarks optimize for reproducibility and automated scoring, they sacrifice ecological validity—the degree to which benchmark tasks reflect real-world usage. B^3 inverts this trade-off: high ecological validity at the cost of reproducibility and automated scoring.

The Generative Separability Gap (Definition 5.11) quantifies this disconnect. Models that score well on SWE-bench but poorly on B^3 have a positive GSG, indicating that their academic benchmark performance overestimates their practical coding capability. This gap is not merely theoretical: it has real economic consequences when organizations select models based on academic benchmarks that poorly predict real-world performance.

9.2 The Single-Evaluator Problem

The primary limitation of B^3 is its dependence on a single evaluator. However, the consistency of Bijan Bowen's evaluation methodology across 50+ videos provides a form of longitudinal reliability that mitigates this concern. The fixed prompt templates,

consistent evaluation environment, and repeated application across models create a de facto standardized protocol, even if formally unstandardized.

Community adoption of B^3 with multiple evaluators would transform it from a practitioner benchmark to a community standard. The multi-evaluator protocol (Section 7.1) provides a path forward.

9.3 Implications for Benchmark Design

B^3 suggests three principles for next-generation LLM benchmarks:

1. **Generative over selective**: Tasks should require complete artifact generation, not answer selection.
2. **Multi-dimensional over single-dimensional**: Benchmarks should span multiple capability dimensions to produce rich profiles.
3. **Ecological validity over reproducibility**: When trade-offs are necessary, tasks that mirror real-world usage should be preferred over tasks optimized for automated scoring.

10 Conclusion

The Bijan Bowen Benchmark (B^3) represents a practitioner-developed evaluation methodology that captures a capability dimension—zero-shot generative coding—poorly measured by existing academic benchmarks. We have formalized its task structure, proven separability, coverage, and monotonicity properties, identified six orthogonal capability dimensions, and introduced the Generative Separability Gap as a measure of ecological validity. The Kimi K3 case study demonstrates B^3's ability to reveal capability profiles that academic benchmarks cannot.

We propose that the LLM evaluation community adopt and extend B^3 with multi-evaluator protocols, temperature-controlled evaluation, and API standardization to transform it from a practitioner benchmark into a community standard. The gap between academic benchmark performance and real-world coding capability is the most important unmeasured dimension in LLM evaluation today, and B^3 provides a principled foundation for closing it.

Acknowledgments

We acknowledge Bijan Bowen for developing the evaluation methodology formalized in this paper. His YouTube channel [6] has provided the LLM community with an invaluable service through consistent, systematic, and honest model evaluation. We also acknowledge the broader practitioner evaluation community whose work motivates this formalization.

References

[1] Moonshot AI. Kimi k3 launch, 2026.

[2] aifor.dev. Bijan bowen – engineer / youtuber. 2026.

[3] Anonymous. Skate: A scalable tournament eval. *OpenReview*, 2025.

[4] Anonymous. Benchmark2: Systematic evaluation of llm benchmarks. *arXiv preprint arXiv:2601.03986*, 2026.

[5] Bijan Bowen. Bijan bowen – ai consultant, 2026.

[6] Bijan Bowen. Bijan bowen youtube channel, 2026.

[7] Bijan Bowen. Kimi k3 is insane – is this a sol & fable competitor?, 2026. YouTube channel: Bijan Bowen.

[8] François Chollet. Arc-agi-2: A new benchmark for abstract reasoning. 2024.

[9] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[10] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *International Conference on Learning Representations (ICLR)*, 2021.

[11] Howard. Rtx 4090 local llm workflow benchmark. 2026.

[12] Carlos E Jimenez, John Yang, Alexander Wheeler, Ayush Naik, Jesse Wetzel, Prabal Bailis, and Yuxin Wang. Swe-bench: Can language models resolve real-world github issues? *International Conference on Learning Representations (ICLR)*, 2024.

[13] kie.ai. What is kimi k3? moonshot's 2.8t, 1m-context flagship, 2026.

[14] GameWorld Project. Gameworld: Towards standardized and verifiable evaluation of multimodal game agents. *ECCV*, 2026.

[15] Expert Review Team. Measuring what matters: Construct validity in large language model benchmarks. *OpenReview*, 2025.

[16] TestingCatalog. Early look at kimi k3 generations from moonshot ai on arena, 2026.

[17] Jihan Yao, Peter Jin, Ke Bao, Qiaolin Yu, Khushi Bhardwaj, Chang Su, Jialei Wang, Yikai Zhu, Sugam Devare, Damon Mosk-Aoyama, Zhen Dong, Venkat Krishna Srinivasan, Yineng Zhang, Oleksii Kuchaiev, Jiantao Jiao, and Banghua Zhu. The measure of all measures: Quantifying llm benchmark quality. *OpenReview*, 2025.