Beyond Static Leaderboards: A Roadmap to Naturalistic, Functional Evaluation of LLMs *

Victor Ojewale[1], **Suresh Venkatasubramanian**[1]

[1]Center for Tech. Responsibility, Brown University, Providence, RI, USA
victor_ojewale@brown.edu

Abstract

Static leaderboards and single turn judgments correlate weakly with deployment outcomes, especially in multilingual and resource constrained settings. This position paper argues that credible evaluation hinges on *verifiability*: ex ante specifications that permit observable checks, repeatable scoring, and auditable evidence. We propose a minimal standard that makes verifiability first class while remaining compatible with existing workflows. The standard has four artifacts: a task schema, a validator entry point, a run card, and required reporting fields. We ground the proposal in prior work on coverage and transparency (Wang et al. 2019; Hendrycks et al. 2021; Liang et al. 2023) and on specification based checks (Chen et al. 2021; Jimenez et al. 2024; Zhou et al. 2023; Ribeiro et al. 2020). We present a prototype evaluation task for schema constrained instruction following with robustness probes and a multilingual protocol, and we attach measurement and governance procedures that link scores to validity arguments. The goal is to replace generic win rates with verifiable claims about task success that better predict real use across languages and contexts.

Introduction

Benchmarks accelerate iteration, yet widely cited scores often fail to predict whether systems succeed on real tasks (Wang et al. 2019; Hendrycks et al. 2021). Holistic work shows that conclusions change once scenario breadth and multi metric reporting are considered (Liang et al. 2023). The missing ingredient is stronger *verifiability*. By verifiability we mean three conditions: success criteria are specified before evaluation, compliance is observable through repeatable checks, and evidence is preserved for audit.

We advocate *naturalistic and functional* evaluation with verifiability at its core. Naturalistic tasks mirror real goals, constraints, and noise. Functional checks determine whether outputs satisfy user relevant specifications. Preference win rates and single turn scores remain useful diagnostics, but they are insufficient when reliability, integration, and equity are at stake. Verifiable functional checks provide clearer signals, especially where data, and expertise are constrained.

*This work was supported in part by a gift from the Heising-Simons Foundation.

Contributions.

- A verifiability first framework for naturalistic, functional evaluation that replaces single scores with interpretable scorecards.
- A minimal, interoperable standard with four artifacts for task exchange and reporting: task schema, validator entry point, run card, and required reporting fields.
- A prototype schema constrained instruction following task with robustness probes and a multilingual protocol to illustrate the standard in practice.
- Measurement and governance protocols that tie results to explicit validity arguments and preserve an auditable evidence trail.

Background and Motivation

Limits of static leaderboards and contamination. Benchmark leaderboards have accelerated iteration, yet single aggregate numbers often fail to predict whether systems accomplish real tasks in deployment (Wang et al. 2019; Hendrycks et al. 2021). Even broad dashboards can mask whether a model satisfied concrete, task-specific requirements (Liang et al. 2023). Overlap between training corpora and evaluation sets further erodes score interpretability and can produce apparent gains without robust capability improvements (Liang et al. 2023; Deng et al. 2024). Recent surveys document contamination pathways in modern LLM pretraining and fine-tuning pipelines and argue for stronger provenance controls and test-set hygiene (Deng et al. 2024).

From preference to verifiable outcomes. Human or LLM preference signals are valuable for fluency and helpfulness, but they can reward surface qualities while overlooking requirement satisfaction, formatting constraints, or brittle reasoning (Zheng et al. 2023). Empirical studies report sensitivity of LLM judges to position and verbosity as well as substantial variance across prompts, seeds, and judge models, motivating randomized presentation, gold calibration, and uncertainty reporting (Jung, Brahman, and Choi 2024; Son et al. 2025). We therefore treat preference as supplemental evidence and place verifiable checks at the center of the measurement strategy.

Naturalistic evaluation lineage. The field has progressed from static, exam-style testing toward naturalistic and functional tasks. Dynabench formalized dynamic, human-in-the-loop collection to preserve difficulty and ecological validity (Kiela et al. 2021). Executable benchmarks such as HumanEval and SWE-bench judge success by running code or resolving real issues end-to-end (Chen et al. 2021; Jimenez et al. 2024). IFEval operationalizes instruction-following through programmatic constraints, while CheckList turns robustness into measurable perturbation families (Zhou et al. 2023; Ribeiro et al. 2020). Concurrently, live or simulated interactive arenas e.g., WildBench-style in-the-wild tasks and web interaction suites seek to approximate user goals and environments (Lin et al. 2024; Chiang et al. 2024). These strands collectively support a shift from average win-rates to outcome-based, verifiable completion of realistic tasks.

Multilingual and underserved contexts. Large cross-lingual evaluations continue to show persistent disparities across languages and varieties (Hu et al. 2020; Clark et al. 2020; Team et al. 2022; Adelani et al. 2021). Code-mixing resources such as GLUECoS, LinCE and CodeMixBench further highlight phenomena that typical English-centric benchmarks ignore (Khanuja et al. 2020; Aguilar, Kar, and Solorio 2020; Yang and Chai 2025). Real deployments in underserved settings often rely on strict schemas, locale-aware number and date formats, and intermittent or low-resource operation. Evaluations should make these constraints explicit and verifiable, report per-language breakdowns with uncertainty, and include code-mix and locale perturbations.

Why verifiability. Putting verifiability first aligns evaluation with responsible measurement practice. Ex ante success criteria reduce researcher degrees of freedom, observable checks and repeatable scoring enable independent reproduction, and preserved artifacts connect scores to evidence for governance uses complementing model cards and datasheets (Mitchell et al. 2019; Gebru et al. 2021). Together, these practices raise the floor on reliability and make claims portable across languages, resource profiles, and tooling stacks.

Naturalistic Functional Evaluation

Naturalistic task settings. A naturalistic evaluation draws tasks from realistic scenarios and preserves their context and messiness. Efforts that curate difficult, user relevant queries captured in live settings surface challenges missed by curated exam style prompts (Liang et al. 2023). Naturalistic tasks may require multi turn clarification, adherence to user specific constraints, and awareness of domain or locale conventions. Evaluations that keep these properties intact better approximate downstream behavior.

Functional success criteria. Functional evaluation associates each task with explicit criteria that define what it means to complete the task(Srivastava et al. 2024; Ojewale, Raji, and Venkatasubramanian 2025). Criteria are expressed as checks that can be verified. In code generation, hidden unit tests provide the decision rule (Chen et al. 2021). In instruction following, verifiable constraints such as length, required phrases, or structural conformance can be checked directly (Zhou et al. 2023). Extending this approach, multilingual functional evaluation demonstrates that these specification-based checks can be applied consistently across languages and resource contexts to test cross-lingual equivalence and schema compliance (Ojewale, Raji, and Venkatasubramanian 2025). Many deployments also require outputs to match strict schemas, for example JSON objects consumed by tools, where an output validator can test parseability, exact keys, and required fields. This shifts evaluation from subjective impressions to verifiable task completion.

Scorecards over single scores. Rather than a single rank, we propose compact scorecards that summarize decision outcomes per task. Each scorecard records whether the model met the specification, how many criteria were satisfied, whether retries were needed, stability under small perturbations (Ribeiro et al. 2020), and an error class label such as syntax, format, logic, retrieval, tool, or safety. Aggregating scorecards across tasks yields success rates and failure mixes while preserving interpretability. This mirrors multi metric reporting norms from HELM while centering specification satisfaction and robustness (Liang et al. 2023). For practitioners, such reports indicate not only how often a model works but also how and why it fails.

A Minimal Standard for Verifiable Evaluation

The standard operationalizes verifiability with four artifacts. It is intentionally small to reduce engineering burden and to interoperate with existing runners.

Artifact 1: Task schema

A machine readable object that declares prompts, inputs, languages, criteria, and perturbations. It describes what to check, not how to optimize.

Artifact 2: Validator entry point

A callable that maps model outputs and optional inputs to criterion level decisions. Checks can be automated or manual. Manual checks require a rubric, a two pass adjudication plan, and agreement reporting.

Artifact 3: Run card

A structured record that enables repeatable scoring and later audit. Include model identifier and version, decoding settings, seeds, judge settings if any, software and hardware, fixture hashes, languages, time stamps, and pointers to raw outputs and validator logs.

Artifact 4: Required reporting fields

Publish at minimum: task success rate, mean specification compliance, invalid output rate, retry distribution, stability under perturbations, and error class distribution. Report per language and per capability family. Include inter rater agreement when humans are involved.

Listing 1: Minimal task schema for schema constrained instruction following.

```
{
  "id": "task.schema_follow.v1",
  "languages": ["en","yo"],
  "prompt": {"turns":[
    {"role":"user",
     "text":"Return a JSON object with
         keys point1, point2, point3 and
         mention 'carbon emissions' at
         least once."}
  ]},
  "schema": {
    "type":"object",
    "required":["point1","point2","
        point3"],
    "properties":{
      "point1":{"type":"string","
          minLength":1},
      "point2":{"type":"string","
          minLength":1},
      "point3":{"type":"string","
          minLength":1}
    },
    "additionalProperties": false
  },
  "criteria": [
    {"name":"json_parse"},
    {"name":"exact_keys"},
    {"name":"keyword_present","args":{"
        phrase":"carbon emissions"}},
    {"name":"non_empty_values"}
  ],
  "perturbations":[
    {"name":"reorder_requirements"},
    {"name":"inject_distractors"},
    {"name":"code_mix_instruction"},
    {"name":"whitespace_variants"}
  ],
  "report": ["success","spec_compliance
      ","retries","stability","
      error_class"]
}
```

Listing 2: Validator output contract with evidence fields.

```
{"success": true,
 "spec_compliance": 1.0,
 "retries": 1,
 "stability": 0.90,
 "checks": {
   "json_parse": true,
   "exact_keys": true,
   "keyword_present": true,
   "non_empty_values": true
 },
 "evidence": {
   "parsed_json":"{...}",
   "keyword_hits":["carbon emissions"]
 },
 "error_class": null}
```

Listing 3: Illustrative run card fields.

```
{
 "model": {"name":"X-Model","version
     ":"2025-09"},
 "decode": {"temperature":0.2,"
     max_tokens":512},
 "seeds": {"sampling":123,"eval":7},
 "env": {"python":"3.11","os":"linux-
     amd64"},
 "fixtures": {"schema_sha256":"...","
     prompt_sha256":"..."},
 "languages": ["en","yo"],
 "judge": null,
 "artifacts": {
   "outputs_path":"runs/2025-10-01/
       outputs.jsonl",
   "logs_path":"runs/2025-10-01/
       validator.log"
 },
 "summary": {"success_rate":0.74,"
     mean_spec":0.88}
}
```

Proposed Prototype Evaluation Task Scenario

Scenario. A product team requires strict JSON outputs from natural language instructions. Downstream services reject malformed structures. Reliability must hold across English, one additional language, and a light code–mix variant. Decision rules are executable (Zhou et al. 2023) and robustness is probed via controlled surface changes (Ribeiro et al. 2020). The task surface and checks are encoded in the *task schema* of Listing 1. The validator's return contract is shown in Listing 2. Each evaluation run is documented with the *run card* in Listing 3 for reproducibility and later audit.

Task Surface

User goal: "Give three brief points that address topic t." **Schema**: exact keys `point1`, `point2`, `point3`; string values(see Listing 1, field `schema`). **Lexical constraint**: phrase p must appear in at least one value (Listing 1, criterion `keyword_present`). **Languages**: `en`, `L2`, and a code–mix prompt that keeps JSON keys in Latin script (Listing 1, fields `languages`, `perturbations`).

Required Behaviors

1. Return a single JSON object that parses under a strict JSON parser (`json_parse` in Listing 1).
2. Use exactly the key set {`point1`, `point2`, `point3`} (`exact_keys`).
3. Satisfy schema types and minimal lengths (`schema_types`, `non_empty_values`).
4. Include the lexical constraint p in at least one value (`keyword_present`).

Validator Entry Point

The deterministic validator maps to criterion booleans and evidence as specified in Listing 2. Success is the conjunction of all criteria. Failures receive an error class from {SYNTAX, KEYS, SCHEMA, LEXICAL, OTHER}. Minimal

evidence snippets that triggered each decision are logged for audit.

Robustness Probes

Controlled perturbations preserve the functional requirement and are enumerated in the `perturbations` field of Listing 1. We instantiate:

1. Reorder bullets in the instruction.
2. Insert short distractor sentences.
3. Whitespace and punctuation variants.
4. Code–mix a fixed list of operators and meta–verbs while freezing JSON keys.
5. Same–language synonym substitutions for content words.

Stability in this regard would be the fraction of perturbed prompts that still pass all checks.

Multilingual and Code–Mix Protocol

Parallel prompts in L2 are produced with forward translation and independent review. A lightweight code–mix substitutes function words while protecting numerals, quoted spans, and schema keys. The same validator applies across `en`, `L2`, and code–mix. Report per–language results to surface disparities (Liang et al. 2023).

Measurement and Reporting

We allow a small retry budget $R \in \{0, 1, 2\}$ under fixed decoding and seeds. Publish a per–language scorecard:

- Success rate (any–pass within R).
- Mean specification compliance (mean of criterion booleans).
- Invalid output rate (strict parse failures).
- Stability under perturbations.
- Retries to success and error–class distribution.

Each run includes a run card as in Listing 3.

Measurement and Governance Protocols

We adopt validity as an argument about score interpretations supported by multiple strands of evidence (Messick 1989; AERA, APA, and NCME 2014). The protocols below make those strands concrete for naturalistic, specification based evaluation and connect them to governance practices.

Content validity. Define the construct precisely and map tasks to capability families. Provide a coverage matrix by language, domain, and interaction pattern. Justify task selection with stakeholder or domain expert review and summarize changes after pilot runs. For multilingual settings, document varieties and code mix types covered and note gaps that remain.

Response process evidence. Describe how tasks, prompts, and translations are produced and checked. For translation, specify forward translation, independent review, and accept–reject rules for edits. For instruction following, publish the logic of each check and examples of borderline cases. If any manual decisions are used, provide rubrics, annotator training materials, and the adjudication workflow.

Internal structure and reliability. Report reliability of checks and stability under perturbations (Ribeiro et al. 2020). Include seeds, prompt variants, and retry budgets in a small factorial design and quantify variance attributable to each factor. Provide bootstrap confidence intervals for success and specification compliance. Track failure modes by error class and report the concentration of errors per task and per language.

Relations to other variables. Assess convergence and divergence with established aggregates such as HELM style metrics (Liang et al. 2023). Where feasible, correlate success with downstream outcomes such as human utility ratings or integration success in a target workflow. If LLM judges are used for qualities that resist hard specifications, report agreement against small gold sets, position randomization, and bias controls (Zheng et al. 2023; Li et al. 2024).

Transparency artifacts. Publish an "evaluation card" that complements model cards and datasheets (Mitchell et al. 2019; Gebru et al. 2021). The card should summarize constructs, task sources, languages, check logic, reliability, uncertainty, and known limitations in one place. Include a short checklist of reproducibility items that must be satisfied for third parties to rerun the suite.

Interoperability and Reproducibility

Align runners with HELM style execution to enable scenario level aggregation and release of prompts and outputs (Liang et al. 2023). Provide seed control and fixture hashes. Containerize validators where possible or publish detailed rubrics and annotation interfaces. Release schemas, validators, and small fixtures under research friendly licenses. For restricted data, publish redacted fixtures and replication instructions, and state constraints clearly.

Discussion and Limitations

Our proposal for naturalistic, functional, and verifiable evaluation is motivated by a simple question: *what does good performance mean when an LLM is actually deployed to do real work?* In deployment, a good model is one that consistently produces useful, correct outputs for the tasks that users need, while avoiding failures that cause harm or break downstream systems. This is not fully captured by today's dominant evaluation signals. Human or LLM preference judgments tend to favor outputs that are eloquent and safe, which is valuable, but they can overlook subtle errors, missed requirements, or brittle reasoning. A user might rate an answer highly because it sounds plausible, even if it is slightly incorrect or not precisely what they needed. Calibration metrics tell us whether a model knows when it does not know, but a well calibrated model can still be uniformly wrong on an entire class of tasks.

Functional evaluation instead targets deployment time behavior by testing models in scenarios that approximate real use and by measuring concrete success. For a coding assistant, what matters is how often a model produces working code that passes tests, not only whether the code looks syntactically plausible. For a customer support assistant, suc-

cess might mean resolving an issue within a small number of turns using the right policies and tools. For multilingual deployments, success includes meeting users in their languages and varieties while respecting strict schemas and locale formats. By constructing evaluations that mirror these realities, we reduce the gap between benchmark performance and practical reliability and avoid the familiar situation where models ace static tests yet disappoint in production.

A key advantage of naturalistic, specification based tasks is that they expose robustness and edge cases. Because each task has explicit criteria and we can vary inputs and conditions, we can interrogate failure modes. Perturbation families, such as adding distractors, rephrasing instructions, or code mixing prompts while holding schemas fixed, make it possible to see where performance collapses despite superficial similarity. Compact scorecards that record success, specification compliance, retries, stability, and error classes provide more actionable feedback than a single scalar. They help distinguish models that fail cleanly and transparently from those that produce confident but subtly non compliant outputs.

At the same time, there are important limitations and risks that temper how far a verifiability first approach can go. A first limitation is the **specifiability ceiling**. Not all valuable capabilities admit crisp, executable criteria. Creativity, cultural nuance, and value alignment often require rubric based or human adjudication, which introduces variability and potential bias even with careful training and agreement checks. LLM judge pipelines increase throughput but remain sensitive to verbosity, position, and self agreement, so residual bias can persist even when combined with functional filters.

A second limitation concerns **gaming and brittleness**. Specification centered evaluation can itself be overfit. Systems may learn to satisfy surface checks without robust competence, for example by overproducing required phrases or rigid JSON shells while still making reasoning errors inside fields. Public fixtures and schemas also increase contamination risk, especially when reused during model training. Perturbation families and multilingual variants reduce these risks but do not eliminate them, and aggregate scorecards can still mask rare but harmful failures that matter disproportionately in high stakes settings.

A third limitation involves **cost, coverage, and reproducibility**. Designing precise checks, building validators, curating naturalistic tasks, and running multilingual pilots impose nontrivial costs. Language and variety coverage will remain incomplete, particularly for code mixing and locale specific phenomena, so uncertainty reporting and explicit gap documentation are necessary. Closed models and rapid versioning further limit exact replication. Run cards, and released fixtures mitigate these issues by making evaluations more transparent, but they cannot fully resolve drift across providers and time.

Taken together, these limitations suggest that naturalistic, functional, and verifiable evaluation should be viewed as a core layer rather than a complete replacement for existing practices. Our position is that specification based checks and scorecards should form the backbone of evaluation, especially where reliability and governance matter, while preference judgments and qualitative analyses remain important complements for aspects that resist formalization. If the community invests in **shared schemas, validators, and multilingual fixtures**, the costs of this approach can be reduced and its benefits made accessible beyond well resourced organizations.

Conclusion

Verifiability is central to credible model evaluation. A small standard that foregrounds ex ante specifications, observable compliance, repeatable scoring, and auditable evidence can raise the floor on reliability, improve multilingual coverage, and make progress claims comparable. We recommend adding specification based checks to existing suites, reporting scorecards rather than single ranks, publishing run cards and evidence artifacts, and using perturbation probes to expose brittleness. The community can build a shared library of schemas, validators, and fixtures that lowers adoption costs and improves trust.

References

Adelani, D. I.; Abbott, J.; Neubig, G.; D'souza, D.; Kreutzer, J.; Lignos, C.; Palen-Michel, C.; Buzaaba, H.; Rijhwani, S.; Ruder, S.; Mayhew, S.; Azime, I. A.; Muhammad, S. H.; Emezue, C. C.; Nakatumba-Nabende, J.; Ogayo, P.; Anuoluwapo, A.; Gitau, C.; Mbaye, D.; Alabi, J.; Yimam, S. M.; Gwadabe, T. R.; Ezeani, I.; Niyongabo, R. A.; Mukiibi, J.; Otiende, V.; Orife, I.; David, D.; Ngom, S.; Adewumi, T.; Rayson, P.; Adeyemi, M.; Muriuki, G.; Anebi, E.; Chukwuneke, C.; Odu, N.; Wairagala, E. P.; Oyerinde, S.; Siro, C.; Bateesa, T. S.; Oloyede, T.; Wambui, Y.; Akinode, V.; Nabagereka, D.; Katusiime, M.; Awokoya, A.; MBOUP, M.; Gebreyohannes, D.; Tilaye, H.; Nwaike, K.; Wolde, D.; Faye, A.; Sibanda, B.; Ahia, O.; Dossou, B. F. P.; Ogueji, K.; DIOP, T. I.; Diallo, A.; Akinfaderin, A.; Marengereke, T.; and Osei, S. 2021. MasakhaNER: Named Entity Recognition for African Languages. *Transactions of the Association for Computational Linguistics*, 9: 1116–1131.

AERA; APA; and NCME. 2014. The Standards for Educational and Psychological Testing.

Aguilar, G.; Kar, S.; and Solorio, T. 2020. LinCE: A Centralized Benchmark for Linguistic Code-switching Evaluation. In Calzolari, N.; Béchet, F.; Blache, P.; Choukri, K.; Cieri, C.; Declerck, T.; Goggi, S.; Isahara, H.; Maegaard, B.; Mariani, J.; Mazo, H.; Moreno, A.; Odijk, J.; and Piperidis, S., eds., *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 1803–1813. Marseille, France: European Language Resources Association. ISBN 979-10-95546-34-4.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H. P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; Ray, A.; Puri, R.; Krueger, G.; Petrov, M.; Khlaaf, H.; Sastry, G.; Mishkin, P.; Chan, B.; Gray, S.; Ryder, N.; Pavlov, M.; Power, A.; Kaiser, L.; Bavarian, M.; Winter, C.; Tillet, P.; Such, F. P.; Cummings, D.; Plappert, M.; Chantzis, F.; Barnes, E.; Herbert-Voss, A.; Guss, W. H.; Nichol, A.; Paino, A.; Tezak, N.; Tang, J.; Babuschkin, I.; Balaji, S.;

Jain, S.; Saunders, W.; Hesse, C.; Carr, A. N.; Leike, J.; Achiam, J.; Misra, V.; Morikawa, E.; Radford, A.; Knight, M.; Brundage, M.; Murati, M.; Mayer, K.; Welinder, P.; McGrew, B.; Amodei, D.; McCandlish, S.; Sutskever, I.; and Zaremba, W. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374.

Chiang, W.-L.; Zheng, L.; Sheng, Y.; Angelopoulos, A. N.; Li, T.; Li, D.; Zhang, H.; Zhu, B.; Jordan, M.; Gonzalez, J. E.; and Stoica, I. 2024. Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. arXiv:2403.04132.

Clark, J. H.; Choi, E.; Collins, M.; Garrette, D.; Kwiatkowski, T.; Nikolaev, V.; and Palomaki, J. 2020. TyDi QA: A Benchmark for Information-Seeking Question Answering in Typologically Diverse Languages. *Transactions of the Association for Computational Linguistics*, 8: 454–470.

Deng, C.; Zhao, Y.; Heng, Y.; Li, Y.; Cao, J.; Tang, X.; and Cohan, A. 2024. Unveiling the Spectrum of Data Contamination in Language Model: A Survey from Detection to Remediation. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 16078–16092. Bangkok, Thailand: Association for Computational Linguistics.

Gebru, T.; Morgenstern, J.; Vecchione, B.; Vaughan, J. W.; Wallach, H.; III, H. D.; and Crawford, K. 2021. Datasheets for datasets. *Commun. ACM*, 64(12): 86–92.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. arXiv:2009.03300.

Hu, J.; Ruder, S.; Siddhant, A.; Neubig, G.; Firat, O.; and Johnson, M. 2020. XTREME: a massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Jimenez, C. E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; and Narasimhan, K. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? arXiv:2310.06770.

Jung, J.; Brahman, F.; and Choi, Y. 2024. Trust or Escalate: LLM Judges with Provable Guarantees for Human Agreement. arXiv:2407.18370.

Khanuja, S.; Dandapat, S.; Srinivasan, A.; Sitaram, S.; and Choudhury, M. 2020. GLUECoS: An Evaluation Benchmark for Code-Switched NLP. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3575–3585. Online: Association for Computational Linguistics.

Kiela, D.; Bartolo, M.; Nie, Y.; Kaushik, D.; Geiger, A.; Wu, Z.; Vidgen, B.; Prasad, G.; Singh, A.; Ringshia, P.; Ma, Z.; Thrush, T.; Riedel, S.; Waseem, Z.; Stenetorp, P.; Jia, R.; Bansal, M.; Potts, C.; and Williams, A. 2021. Dynabench: Rethinking Benchmarking in NLP. In Toutanova, K.; Rumshisky, A.; Zettlemoyer, L.; Hakkani-Tur, D.; Beltagy, I.; Bethard, S.; Cotterell, R.; Chakraborty, T.; and Zhou, Y., eds., *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4110–4124. Online: Association for Computational Linguistics.

Li, H.; Dong, Q.; Chen, J.; Su, H.; Zhou, Y.; Ai, Q.; Ye, Z.; and Liu, Y. 2024. LLMs-as-Judges: A Comprehensive Survey on LLM-based Evaluation Methods. arXiv:2412.05579.

Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A.; Newman, B.; Yuan, B.; Yan, B.; Zhang, C.; Cosgrove, C.; Manning, C. D.; Ré, C.; Acosta-Navas, D.; Hudson, D. A.; Zelikman, E.; Durmus, E.; Ladhak, F.; Rong, F.; Ren, H.; Yao, H.; Wang, J.; Santhanam, K.; Orr, L.; Zheng, L.; Yuksekgonul, M.; Suzgun, M.; Kim, N.; Guha, N.; Chatterji, N.; Khattab, O.; Henderson, P.; Huang, Q.; Chi, R.; Xie, S. M.; Santurkar, S.; Ganguli, S.; Hashimoto, T.; Icard, T.; Zhang, T.; Chaudhary, V.; Wang, W.; Li, X.; Mai, Y.; Zhang, Y.; and Koreeda, Y. 2023. Holistic Evaluation of Language Models. arXiv:2211.09110.

Lin, B. Y.; Deng, Y.; Chandu, K.; Brahman, F.; Ravichander, A.; Pyatkin, V.; Dziri, N.; Bras, R. L.; and Choi, Y. 2024. WildBench: Benchmarking LLMs with Challenging Tasks from Real Users in the Wild. arXiv:2406.04770.

Messick, S. 1989. Validity. In Linn, R. L., ed., *Educational Measurement*, 13–103. New York, NY: American Council on education and Macmillan: Macmillan, 3 edition.

Mitchell, M.; Wu, S.; Zaldivar, A.; Barnes, P.; Vasserman, L.; Hutchinson, B.; Spitzer, E.; Raji, I. D.; and Gebru, T. 2019. Model Cards for Model Reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, 220–229. New York, NY, USA: Association for Computing Machinery. ISBN 9781450361255.

Ojewale, V.; Raji, I. D.; and Venkatasubramanian, S. 2025. Multi-lingual Functional Evaluation for Large Language Models. arXiv:2506.20793.

Ribeiro, M. T.; Wu, T.; Guestrin, C.; and Singh, S. 2020. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4902–4912. Online: Association for Computational Linguistics.

Son, S.; Oh, J.-M.; Jin, H.; Jang, C.; Jeong, J.; and Kim, K. 2025. Arena-Lite: Efficient and Reliable Large Language Model Evaluation via Tournament-Based Direct Comparisons. arXiv:2411.01281.

Srivastava, S.; B, A. M.; V, A. P.; Menon, S.; Sukumar, A.; T, A. S.; Philipose, A.; Prince, S.; and Thomas, S. 2024. Functional Benchmarks for Robust Evaluation of Reasoning Performance, and the Reasoning Gap. arXiv:2402.19450.

Team, N.; Costa-jussà, M. R.; Cross, J.; Çelebi, O.; Elbayad, M.; Heafield, K.; Heffernan, K.; Kalbassi, E.; Lam, J.; Licht, D.; Maillard, J.; Sun, A.; Wang, S.; Wenzek, G.; Youngblood, A.; Akula, B.; Barrault, L.; Gonzalez, G. M.; Hansanti, P.; Hoffman, J.; Jarrett, S.; Sadagopan, K. R.; Rowe, D.; Spruit, S.; Tran, C.; Andrews, P.; Ayan, N. F.; Bhosale, S.; Edunov, S.; Fan, A.; Gao, C.; Goswami, V.; Guzmán, F.; Koehn, P.; Mourachko, A.; Ropers, C.; Saleem, S.; Schwenk, H.; and Wang, J. 2022. No Language

Left Behind: Scaling Human-Centered Machine Translation. arXiv:2207.04672.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. *SuperGLUE: a stickier benchmark for general-purpose language understanding systems*. Red Hook, NY, USA: Curran Associates Inc.

Yang, Y.; and Chai, Y. 2025. CodeMixBench: Evaluating Code-Mixing Capabilities of LLMs Across 18 Languages. In Christodoulopoulos, C.; Chakraborty, T.; Rose, C.; and Peng, V., eds., *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2139–2169. Suzhou, China: Association for Computational Linguistics. ISBN 979-8-89176-332-6.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23. Red Hook, NY, USA: Curran Associates Inc.

Zhou, J.; Lu, T.; Mishra, S.; Brahma, S.; Basu, S.; Luan, Y.; Zhou, D.; and Hou, L. 2023. Instruction-Following Evaluation for Large Language Models. arXiv:2311.07911.