Ranking and Classifying AI Benchmarks

Reece Shiraishi, Cornell University; Benjamin Hawks, FNAL – FERMILAB-POSTER-25-0117-STUDENT

Introduction

Despite the rapidly expanding projects on AI benchmarking, there is no universally accepted way to exactly determine the quality of a benchmark. The term “AI benchmark” is poorly defined and understood, further complicating precise judgments.

Purpose

This project seeks to provide criteria for determining benchmark quality. Assisted by a Python script, researchers will have the ability to compare benchmarks using a single definition and rating scale.

Benchmark Visualization

The MLCommons team previously compiled a list of 23 benchmarks using AI tools. Each benchmark was manually verified, then written in a machine-readable YAML format.

A Python script was written to convert the YAML into Markdown and TeX tables. With command-line flags, script users can dictate whether the script puts each entry in a separate file, omits the provided BibTeX citation, or writes the table into a standalone TeX file.

```python
def _load_single_yaml_file(self, file_path: str, enable_error_messages: bool = True) -> list[dict]:
        list of benchmark entries for the YAML file
    """
    try:
        with open(file_path, 'r', encoding='utf-8') as f:
            content = yaml.safe_load(f)

        if isinstance(content, dict):
            return [content]

        elif isinstance(content, list):
            return content

        else:
            raise ValueError("Unsupported YAML format. Expected a dict or list of dicts.")

    except yaml.YAMLError as e:
        if enable_error_messages:
            print(f'{_RED}YAML syntax error in "{file_path}": \n{e}{_DEFAULT_COLOR}')
        return []

def _load_multiple_yaml_files(self, file_paths: list[str], enable_error_messages: bool = True) -> list[dict]:
    """
    Returns a list of dictionaries representing the contents of a YAML file.

    Parameters:
        file_paths (list[str]): list of file paths to load from
        enable_error_messages (bool): whether to print error messages to the console. Default True
    Returns:
        contents of given YAML files
    """
    records = []
    for path in file_paths:
        records += self._load_single_yaml_file(path, enable_error_messages)
    return records

def _load_yamls(self, file_paths: str | list[str], overwriting_contents: bool = True, print_syntax_errors: bool = True) -> None:
    """
    Loads the contents of `file_paths` into this YamlManager.

    If `overwriting_prev` is True, any previous contents are overwritten upon load.

    Raises a FileNotFoundError if an invalid filepath is given.
    If a YAML file contains syntax errors, the file is **not loaded**.

    Parameters:
        file_paths (str or list[str]): filepath(s) to load from
        overwriting_contents (bool, default=True): whether to overwrite the manager's existing contents.
        print_syntax_errors (bool, default=True): whether to print error messages upon finding YAML syntax errors.
    """
    if isinstance(file_paths, str):
        if overwriting_contents:
            self._yaml_dicts = self._load_multiple_yaml_files([file_paths], print_syntax_errors)
        else:
            self._yaml_dicts += self._load_multiple_yaml_files([file_paths], print_syntax_errors)
```

Source code snippet of the script

```yaml
description: |
  The date of availability of the benchmark. If an official release date is
  not available, use the date of adding the entry.
condition: required

version: v1.0
description: |
  The version number of the benchmark
condition: optional

last_updated: '2024-05-01'  # Could also be null
description: |
  The date when the entry was last updated. The format is YYYY-mm-dd
condition: optional

expired: null  # This may be deprecated or duplicate the valid field
description: |
  An indication if the benchmark is no longer valid
condition: optional

valid: true
date: '2025-01-01'
description: |
  Identifies if the benchmark is valid at the time of review. Invalidity may be caused by the code or data
  no longer being available.
condition: required

name: Jet Classification
description: |
  The name of the benchmark
condition: required

url: https://github.com/fastmachinelearning/fastml-science/tree/main/jet-classify
description: |
  The main URL for this benchmark. We also recommend adding it to the citation.
condition: required

doi: xyz...
description: |
```

Part of a YAML entry for a benchmark

Throughout the development process, the YAML format and required attributes changed often. For greater flexibility, the script was modified to handle YAMLs in any data arrangement.

Date	Expiration	Valid	Name	URL	Domain	Focus	Keywords	Description	Task Types	AI Capability	Metrics	Models	Notes	Citation	Problem Specification Rating	Problem Specification Reason	Dataset Rating	Dataset Reason	Ma Rat
2020-09-07		yes	MMLU Massive Multitask Language Understanding	https://paperswithcode.com/dataset/mmlu	Multidomain	Academic knowledge and reasoning across 57 subjects	multitask, multiple-choice, zero-shot, few-shot, knowledge probing	Measuring Massive Multitask Language Understanding MMLU is a benchmark of 57 multiple-choice tasks covering elementary mathematics, US history, computer science, law, and more, designed to evaluate a model's breadth and depth of knowledge in zero-shot and few-shot settings.	Multiple choice	General reasoning, subject-matter understanding	Accuracy	GPT-4o, Gemini 1.5 Pro, o1, DeepSeek-R1	Good	MMLU (Massive Multitask Language Understanding)	6	TO DO	6.5	TO DO	10
2023-11-20		yes	GPQA Diamond	https://arxiv.org/abs/2311.12022	Science	Graduate-level scientific reasoning	Google-proof, graduate-level, science QA, chemistry, physics	GPQA is a dataset of 448 challenging, multiple-choice questions in biology, physics, and chemistry, written by domain experts. It is "Google-proof"—experts score 65% -74% after error correction while skilled non-experts with web access score only 34%. Models like GPT-4 and Llama like GPT-4 reach around 39% accuracy.	Multiple choice, Multi-step QA	Scientific reasoning, deep knowledge	Accuracy	o1, DeepSeek-R1	Good	GPQA Diamond	6.5	Good description of how the problems are received, but little information on how the models are tested	8.5	Easily able to access dataset, No labels or train/test/valid split	10
2018-03-14		yes	ARC-Challenge Advanced Reasoning Challenge	https://allenai.org/data/arc	Science	Grade-school science with reasoning emphasis	grade-school, science QA, challenge set, reasoning	The AI2 Reasoning Challenge ARC Challenge set comprises 7,787 genuine grade-school science questions that retrieval-based and word co-occurrence algorithms both fail, requiring advanced reasoning over a 14-million-sentence corpus.	Multiple choice	Commonsense and scientific reasoning	Accuracy	GPT-4, Claude	Good	ARC-Challenge (Advanced Reasoning Challenge)	9	Exact format of data, questions, and answers are specified. No HW constraints	10	Data accessible, with instructions on how to download the data via CLI tools	10
2025-01-24		yes	Humanity's Last Exam	https://arxiv.org/abs/2501.14249	Multidomain	Broad cross-domain academic reasoning	cross-domain, academic exam, multiple-choice, multidisciplinary	Humanity's Last Exam is a multi-domain, multiple-choice benchmark containing 2,000 questions across dozens of academic disciplines, designed to evaluate LLMs' ability to reason across domains without	Multiple choice	Cross-domain academic reasoning	Accuracy		Good	Humanity's Last Exam	8.5	Format of inputs (natural language) and outputs (multiple choice, short natural language) specified. No HW constraints	6	Data accessible through Hugging Face, but unclear on going contact information to access	10

Markdown output of the Python script

Rating System Development

MLCommons developed 6 key points on the definition of a ML benchmark.

- Problem specification: Well-defined task and format
- Dataset: Publicly accessible and downloadable data
- Metrics: Reproducible and quantitative comparison method
- Reference solution: At least one model ran with the benchmark
- Documentation: Good explanations for the above
- Software (added later, formerly part of Documentation): Easily followed instructions for downloading and running

For each of the qualities, a rubric was created to objectively rank benchmarks on how well they match each quality. Scores for each category range from 0 (lowest; nonexistent) to 10 (highest; perfectly matches criteria).

4. Reference Solution

Baseline implementation demonstrating task feasibility.

- **9–10**: Well documented, reproducible baseline provided with performance results.
- **7–8**: Functional baseline included and evaluated, but lacks full documentation or reproducibility details.
- **5–6**: Baseline exists but is not evaluated using the benchmark or has little instructions for reproduction
- **3–4**: Baseline performance briefly mentioned but not run with the benchmark, or partial implementation with no reported results.
- **1–2**: Reference solution briefly mentioned but is vague
- **0:** No reference solution or baseline presented

Sample rubric for the “reference solution” category

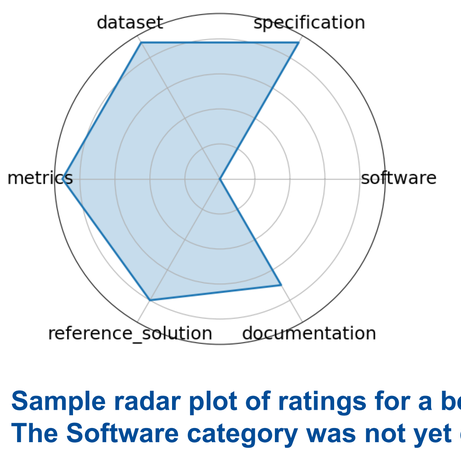
6 Categories based on benchmark definition, all range from 0-5:

- Reference Solution

  Is a reference solution available? Is it well documented?
  Does it list all hardware & software requirements?
  Are all metrics evaluated? Is the baseline model trainable/open?
  (Score = Sum of the above where answer is yes)

Updated rubric for the “reference solution” category, as modified by the MLCommons team

Using the created rubric, each benchmark was evaluated. Ratings were later added to the YAML files. The Python script was updated to include the ratings in its output table.



Jet Classification

Link: https://github.com/fastmachinelearning/fastml-science/tree/main/jet-classify

Problem Spec: 8 — Classification is clearly defined for real-time inference on simulated LHC jets. Input features (HLFs) are documented, though exact latency or resource constraints are not numerically specified.

Dataset: 9 — Two datasets (OpenML and Zenodo) are public, well-formatted, and documented; FAIR principles are followed, though richer metadata would raise confidence to a 10.

Performance Metrics: 9 — AUC and Accuracy are standard, quantitative, and well-aligned with goals of jet tagging and inference efficiency.

Reference Solution: 8 — Float and quantized Keras/QKeras models are provided with results. Reproducibility is good, though full automation and documentation could be improved.

Documentation: 8 — GitHub contains baseline code, data loaders, and references, but setup for deployment (e.g., FPGA pipeline) requires familiarity with the tooling.

Sample radar plot of ratings for a benchmark and accompanying explanations. The Software category was not yet evaluated at the time of writing.

This manuscript has been authored by Fermi Forward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships Program (SULI).