Statistical multi-metric evaluation and visualization of LLM system predictive performance

IBM **Research**

IBM **Research** Israel

Samuel Ackerman (samuel.ackerman@ibm.com), Eitan Farchi, Orna Raz, and Assaf Toledo

Motivation

- Rapid release of new versions of large language models (LLMs).
- Performance is typically assessed by **leaderboards**, without statistical comparisons: Is the difference between the top two models' averages large or small?
- Metric aggregation may be done naively (e.g., min-max rescaling).
- *Statistical testing is not trivial to do correctly*, and the potential audience—especially in industry—who would use it for model evaluation do not generally have the know-how to do so.

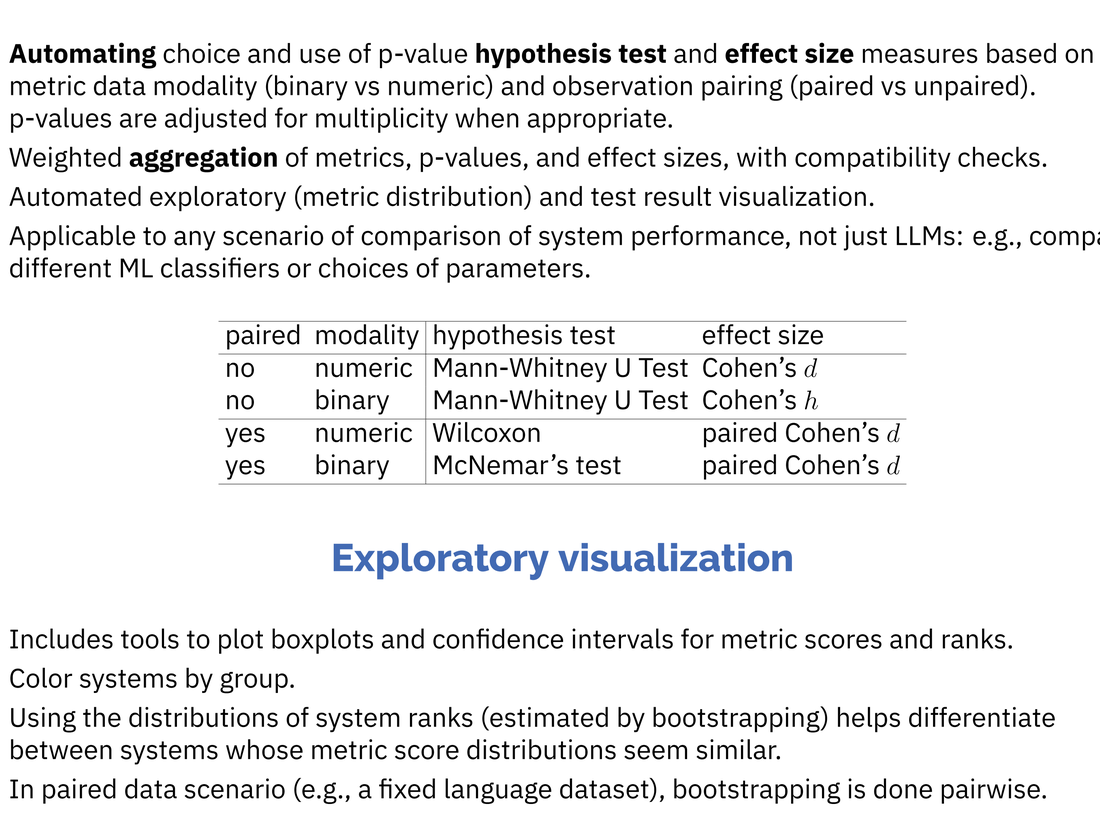
systems

metrics

T	Model	Average	IFEval	BBH	MATH Lvl 5	GPQA	MUSR	MMLU-PRO	CO_2 cost (kg)
	mistralai/Mixtral-8x22B-Instruct-v0.1	33.89	71.04	44.11	18.73	16.44	13.49	38.7	47.15
	CohereForAI/c4ai-command-r-plus-08-2024	33.58	75.4	42.84	12.01	13.42	19.84	38.01	22.32
	jpacifico/Chocolatine-14B-Instruct-DPO-v1.2	33.54	68.52	49.85	19.41	10.07	12.35	41.07	1.54
	tanliboy/lambda-qwen2.5-14b-dpo-test	33.52	82.31	48.45	0	14.99	12.59	42.75	1.8
	Goekdeniz-Guelmez/Josiefied-Qwen2.5-14B-Instruct-abliterated-v4	33.51	82.92	48.05	0	12.3	13.15	44.65	1.75
	TheTsar1209/qwen-carpmuscle-v0.2	33.48	52.57	48.18	27.19	14.09	12.75	46.08	2.25
	migtissera/Tess-v2.5.2-Qwen2-72B	33.28	44.94	52.31	27.42	13.42	10.89	50.68	14.61
	alpindale/WizardLM-2-8x22B	32.98	52.72	48.58	24.55	17.56	14.54	39.96	93.31
	TheTsar1209/qwen-carpmuscle-v0.1	32.92	56.22	48.83	23.11	12.53	10.15	46.67	2.18
	microsoft/Phi-3-medium-4k-instruct	32.9	64.23	49.38	18.35	11.52	13.05	40.84	1.46
	01-ai/Yi-1.5-34B-Chat	32.89	60.67	44.26	24.92	15.32	13.06	39.12	11.21

Introductory example

- CrossCodeEval ([2]) contains different LLM **systems**' code predictions for different languages, with multiple evaluation **metrics** (edit similarity, exact match, F-1, precision, recall).
- Goal: Using a single metric or a (weighted) aggregate of them, statistically test whether one LLM system performs differently than another on a given language **dataset**.
- **Aggregation**: is system 1 [metric] better than system 2 *across language datasets*?



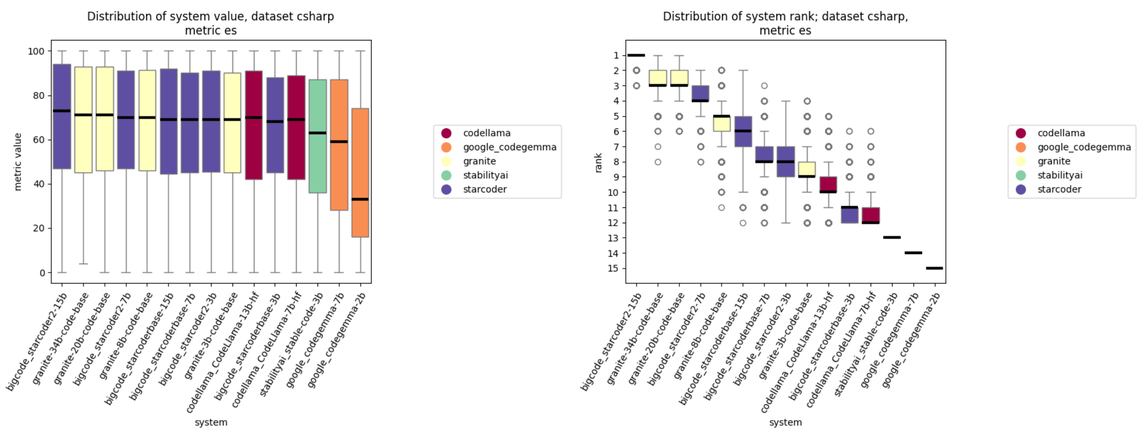
Our contribution

- **Automating** choice and use of p-value **hypothesis test** and **effect size** measures based on metric data modality (binary vs numeric) and observation pairing (paired vs unpaired). p-values are adjusted for multiplicity when appropriate.
- Weighted **aggregation** of metrics, p-values, and effect sizes, with compatibility checks.
- Automated exploratory (metric distribution) and test result visualization.
- Applicable to any scenario of comparison of system performance, not just LLMs: e.g., compare different ML classifiers or choices of parameters.

paired	modality	hypothesis test	effect size
no	numeric	Mann-Whitney U Test	Cohen's d
no	binary	Mann-Whitney U Test	Cohen's h
yes	numeric	Wilcoxon	paired Cohen's d
yes	binary	McNemar's test	paired Cohen's d

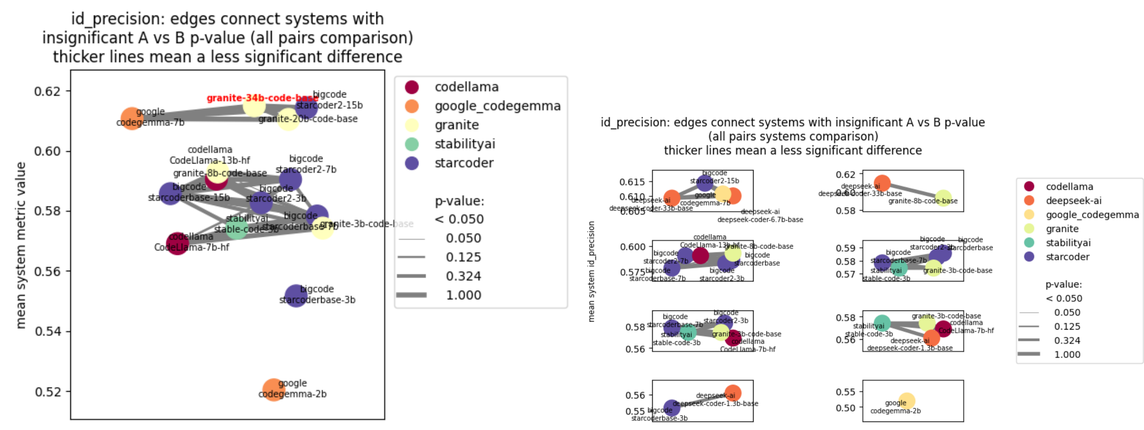
Exploratory visualization

- Includes tools to plot boxplots and confidence intervals for metric scores and ranks.
- Color systems by group.
- Using the distributions of system ranks (estimated by bootstrapping) helps differentiate between systems whose metric score distributions seem similar.
- In paired data scenario (e.g., a fixed language dataset), bootstrapping is done pairwise.



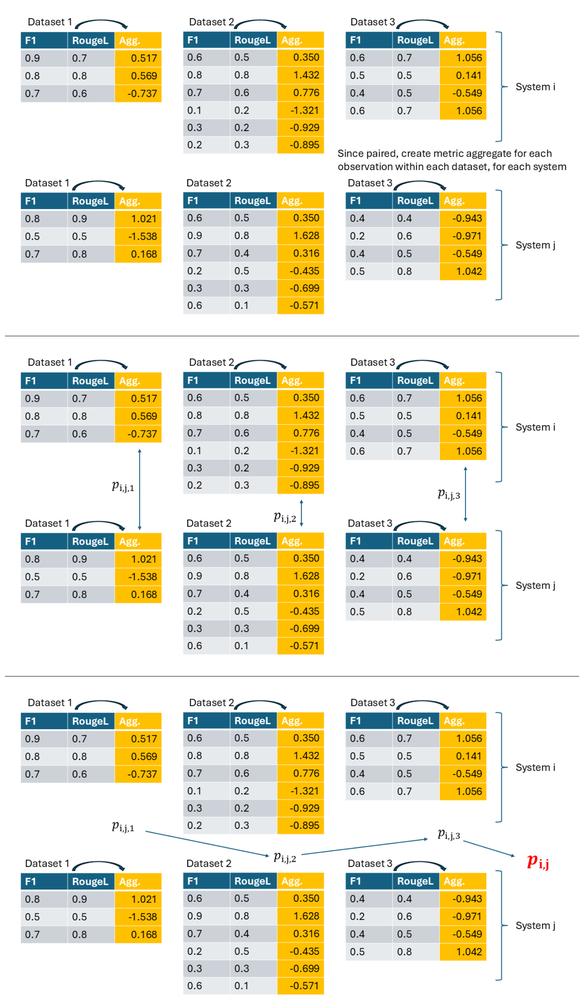
Visualization of pairwise test results

- An original and novel visualization method.
- Graph **vertices** correspond to systems; vertical coordinate is system's metric average.
- An **edge** is drawn between a pair of systems that are statistically compared if the difference is *not statistically significant* (either p-value or effect size); if significant, the edge is missing. A thicker edge represents *a less-significant difference* (higher p-value, lower effect size).
- If p-values $p_{i,j}$ were not cross-dataset aggregates, they are adjusted for multiplicity (FDR, [1]).
- A graph **clique** represents a group of systems whose performance is *mutually not statistically significantly different*. We show all cliques of at least size k (default 2).



Schematic of analysis

For a given pair $(i,\ j)$ of systems to compare:



Step 1 (optional):

- Standardize each system's metrics using the pooled metric values for all systems, to preserve system quality order.
- For each system, create an aggregate metric by (weighted) average across the rows of the standardized columns in each dataset.

Step 2:

- Within each dataset, each system pair $(i,\ j)$ of interest can be compared using a paired observation test on the metric aggregate.
- Test returns either p-value or effect size.

Step 3:

- For each system pair (i, j), further **aggregate** the (weighted) dataset-wise p-values or effect sizes to give an overall measure of system difference.
- P-values: Wilson's harmonic mean ([5],[4]).
- Effect sizes: Inverse-variance weighting ([3]).

Code example

```
# create a paired or unpaired testing object for a given set of systems
paired_tester = PairedDifferenceTest(system_names=systems_names_list)
metrics = ["es", "id_precision", "id_recall"]
# format data into a (paired) collection of score sample lists
score_sample_list_collection =
    [paired_tester.format_as_samples(samples_list=[lang_df[system][mm].to_numpy()
                                                   for system in system_names_list],
                                     dataset_name="python", higher_is_better=True, metric=mm)
                for mm in metrics]

# exploratory visualization
paired_tester.plot_system_metric_distributions(samples_list=score_sample_list_collection[0])
# paired sample lists can be directly aggregated into a new sample list
paired_tester.standardize_metrics_aggregate(samples_lists=score_sample_list_collection, metric_weights=[1,2,3])

# list of significance testing reports, one for each metric
# pass individual reports to visualization
# pass list of reports for p-value/effect size aggregation,
#  and possibly further visualization
score_sample_reports = [paired_tester.signif_pair_diff(score_sample_list)
                        for score_sample_list in score_sample_list_collection]
# aggregate reports
paired_tester.aggregate_signif_report_list(score_sample_reports)
# visualize reports
paired_tester.metric_significant_pairs_graph(test_res=score_sample_reports[0])
```

References

[1] Yoav Benjamini and Daniel Yekutieli. "The control of the false discovery rate in multiple testing under dependency". In: *Annals of Statistics* (2001), pp. 1165–1188.
[2] Yangruibo Ding et al. "CrossCodeEval: A diverse and multilingual benchmark for cross-file code completion". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 46701–46723.
[3] Herbert M Turner III and Robert M Bernard. "Calculating and synthesizing effect sizes". In: *Contemporary Issues in Communication Science and Disorders* 33. Spring (2006), pp. 42–55.
[4] Daniel J Wilson. *harmonicmeanp tutorial*. 2024. URL: https://cran.r-project.org/web/packages/harmonicmeanp/vignettes/harmonicmeanp.html.
[5] Daniel J Wilson. "The harmonic mean p-value for combining dependent tests". In: *Proceedings of the National Academy of Sciences* 116.4 (2019), pp. 1195–1200.

Paper:

Book: