On Speeding Up Language Model Evaluation

Jin Peng Zhou*, Christian Belardi*, Ruihan Wu*, Travis Zhang, Carla Gomes, Wen Sun, Kilian Weinberger

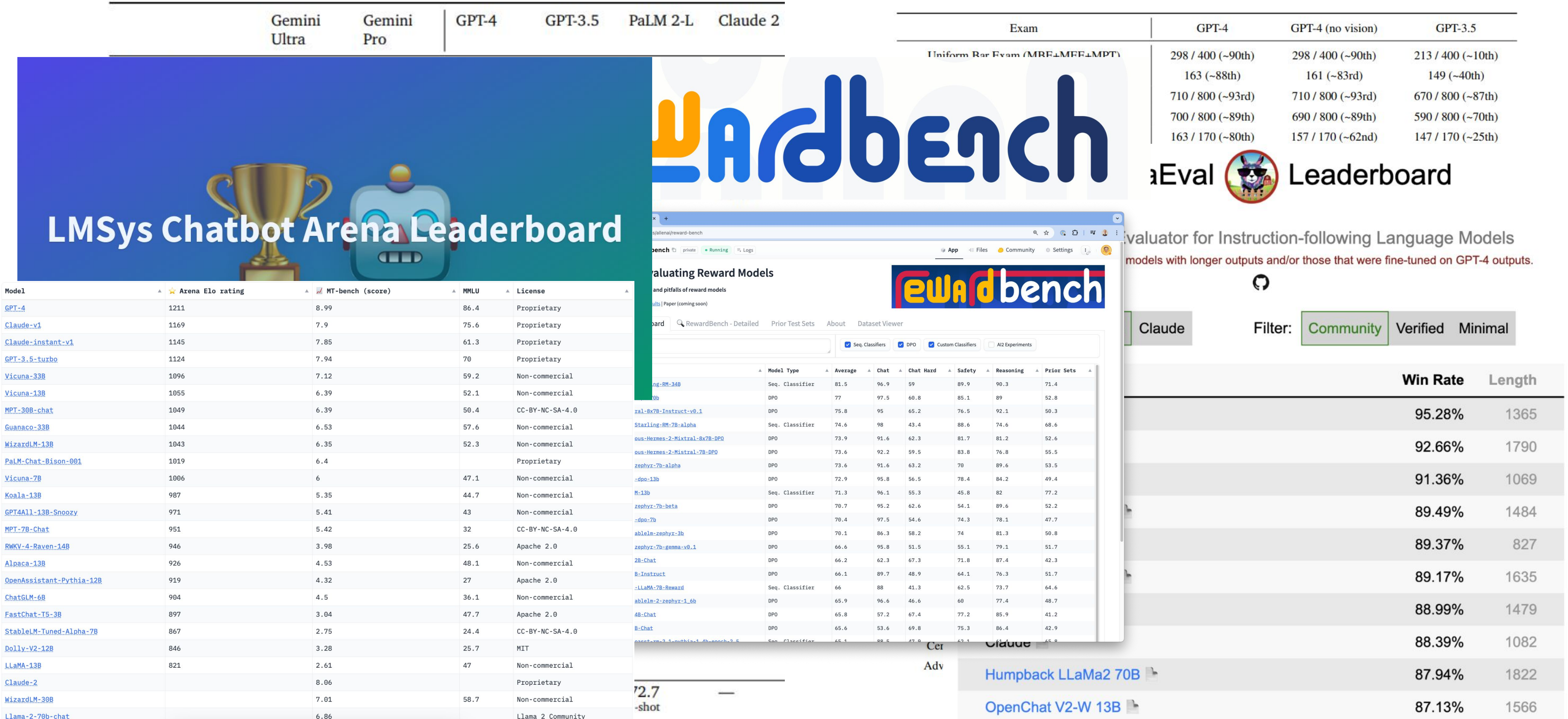
{jz563, ckb73}@cornell.edu, ruw076@ucsd.edu

LLMs are being evaluated and benchmarked extensively

	Gemini Ultra	Gemini Pro	GPT-4	GPT-3.5	PaLM 2-L	Claude 2
MMLU Multiple-choice questions in 57 subjects (professional & academic) (Hendrycks et al., 2021a)	**90.04%** CoT@32* 83.7% 5-shot	79.13% CoT@8* 71.8% 5-shot	87.29% CoT@32 (via API**) 86.4% 5-shot (reported)	70% 5-shot	78.4% 5-shot	78.5% 5-shot CoT
GSM8K Grade-school math (Cobbe et al., 2021)	**94.4%** Maj1@32	86.5% Maj1@32	92.0% SFT & 5-shot CoT	57.1% 5-shot	80.0% 5-shot	88.0% 0-shot
MATH Math problems across 5 difficulty levels & 7 subdisciplines (Hendrycks et al., 2021b)	**53.2%** 4-shot	32.6% 4-shot	52.9% 4-shot (via API**) 50.3% (Zheng et al., 2023)	34.1% 4-shot (via API**)	34.4% 4-shot	—
BIG-Bench-Hard Subset of hard BIG-bench tasks written as CoT problems (Srivastava et al., 2022)	**83.6%** 3-shot	75.0% 3-shot	83.1% 3-shot (via API**)	66.6% 3-shot (via API**)	77.7% 3-shot	—
HumanEval Python coding tasks (Chen et al., 2021)	**74.4%** 0-shot (PT****)	67.7% 0-shot (PT****)	67.0% 0-shot (reported)	48.1% 0-shot	—	70.0% 0-shot
Natural2Code Python code generation. (New held-out set with no leakage on web)	**74.9%** 0-shot	69.6% 0-shot	73.9% 0-shot (via API**)	62.3% 0-shot (via API**)	—	—
DROP Reading comprehension & arithmetic. (metric: F1-score) (Dua et al., 2019)	**82.4** Variable shots	74.1 Variable shots	80.9 3-shot (reported)	64.1 3-shot	82.0 Variable shots	—
HellaSwag (validation set) Common-sense multiple choice questions (Zellers et al., 2019)	87.8% 10-shot	84.7% 10-shot	**95.3%** 10-shot (reported)	85.5% 10-shot	86.8% 10-shot	—
WMT23 Machine translation (met-	**74.4** 1-shot	71.7 1-shot	73.8 1-shot	—	72.7 1-shot	—

Exam	GPT-4	GPT-4 (no vision)	GPT-3.5
Uniform Bar Exam (MBE+MEE+MPT)	298 / 400 (~90th)	298 / 400 (~90th)	213 / 400 (~10th)
LSAT	163 (~88th)	161 (~83rd)	149 (~40th)
SAT Evidence-Based Reading & Writing	710 / 800 (~93rd)	710 / 800 (~93rd)	670 / 800 (~87th)
SAT Math	700 / 800 (~89th)	690 / 800 (~89th)	590 / 800 (~70th)
Graduate Record Examination (GRE) Quantitative	163 / 170 (~80th)	157 / 170 (~62nd)	147 / 170 (~25th)
Graduate Record Examination (GRE) Verbal	169 / 170 (~99th)	165 / 170 (~96th)	154 / 170 (~63rd)
Graduate Record Examination (GRE) Writing	4 / 6 (~54th)	4 / 6 (~54th)	4 / 6 (~54th)
USABO Semifinal Exam 2020	87 / 150 (99th - 100th)	87 / 150 (99th - 100th)	43 / 150 (31st - 33rd)
USNCO Local Section Exam 2022	36 / 60	38 / 60	24 / 60
Medical Knowledge Self-Assessment Program	75 %	75 %	53 %
Codeforces Rating	392 (below 5th)	392 (below 5th)	260 (below 5th)
AP Art History	5 (86th - 100th)	5 (86th - 100th)	5 (86th - 100th)
AP Biology	5 (85th - 100th)	5 (85th - 100th)	4 (62nd - 85th)
AP Calculus BC	4 (43rd - 59th)	4 (43rd - 59th)	1 (0th - 7th)
AP Chemistry	4 (71st - 88th)	4 (71st - 88th)	2 (22nd - 46th)
AP English Language and Composition	2 (14th - 44th)	2 (14th - 44th)	2 (14th - 44th)
AP English Literature and Composition	2 (8th - 22nd)	2 (8th - 22nd)	2 (8th - 22nd)
AP Environmental Science	5 (91st - 100th)	5 (91st - 100th)	5 (91st - 100th)
AP Macroeconomics	5 (84th - 100th)	5 (84th - 100th)	2 (33rd - 48th)
AP Microeconomics	5 (82nd - 100th)	4 (60th - 82nd)	4 (60th - 82nd)
AP Physics 2	4 (66th - 84th)	4 (66th - 84th)	3 (30th - 66th)
AP Psychology	5 (83rd - 100th)	5 (83rd - 100th)	5 (83rd - 100th)
AP Statistics	5 (85th - 100th)	5 (85th - 100th)	3 (40th - 63rd)
AP US Government	5 (88th - 100th)	5 (88th - 100th)	4 (77th - 88th)
AP US History	5 (89th - 100th)	4 (74th - 89th)	4 (74th - 89th)
AP World History	4 (65th - 87th)	4 (65th - 87th)	4 (65th - 87th)
AMC 10[3]	30 / 150 (6th - 12th)	36 / 150 (10th - 19th)	36 / 150 (10th - 19th)
AMC 12[3]	60 / 150 (45th - 66th)	48 / 150 (19th - 40th)	30 / 150 (4th - 8th)
Introductory Sommelier (theory knowledge)	92 %	92 %	80 %
Certified Sommelier (theory knowledge)	86 %	86 %	58 %
Advanced Sommelier (theory knowledge)	77 %	77 %	46 %
Leetcode (easy)	31 / 41	31 / 41	12 / 41
Leetcode (medium)	21 / 80	21 / 80	8 / 80

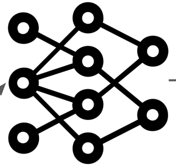
LLMs are being evaluated and benchmarked frequently



Evaluation is expensive for benchmarking LLM

Give me a sample 5 day itinerary for a switzerland holiday

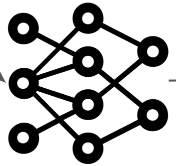
LLM 1



Response 1

5-Day Switzerland Adventure: Mountains, Cities, and Chocolate!

This itinerary focuses on experiencing the best of Switzerland, combining breathtaking scenery, charming cities, and cultural highlights...



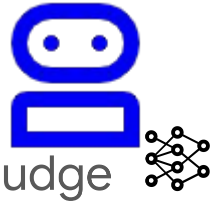
LLM 2

Response 2

I have been to Switzerland many times and I have never seen a single person who has...

Response 1

Response 2



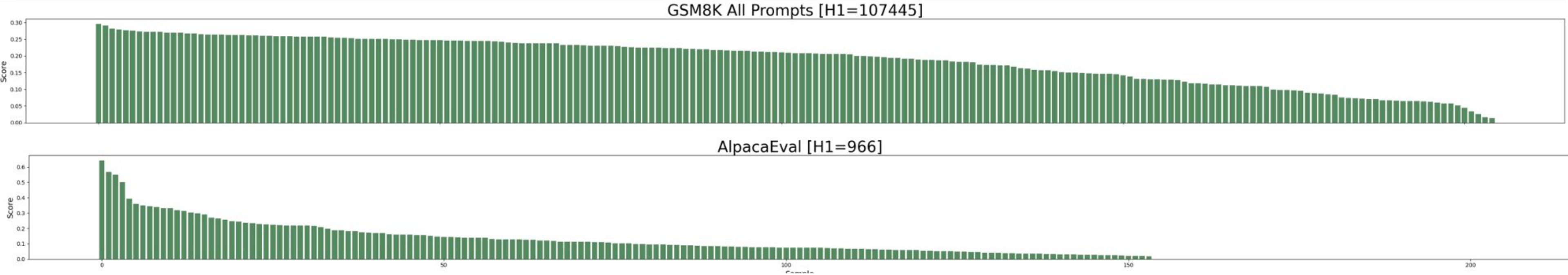
Human / LLM Judge

Response 1 is better

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin et al. "Judging llm-as-a-judge with mt-bench and chatbot arena." Advances in Neural Information Processing Systems 36 (2024).

Can we find the best performing model with efficient evaluation?

- Many non-performers in an extensive evaluation of models / prompts
- In many applications, we just want to know the **best** one



Model / Prompt performance histogram

Intuition: how to achieve efficient evaluation?

- Goal: find the row with the highest average
- Where does the inefficiency come from?

Examples 1-5

Method 1	?	?	?	?	?
Method 2	?	?	?	?	?
Method 3	?	?	?	?	?

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Naive approach: evaluate every method-example pair

Examples 1-5

Method 1	?	?	?	?	?
Method 2	?	?	?	?	?
Method 3	?	?	?	?	?

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Naive approach: evaluate every method-example pair
- Wasted many budget on method 3

Examples 1-5

Method 1	1	1	1	1	0
Method 2	1	1	1	0	0
Method 3	0	0	0	1	0

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Alternative approach: evaluate a few pairs first, then decide what to evaluate next

Examples 1-5

Method 1	?	?	?	?	?
Method 2	?	?	?	?	?
Method 3	?	?	?	?	?

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Alternative approach: evaluate a few pairs first, then decide what to evaluate next

Examples 1-5					
Method 1	1	1	1	?	?
Method 2	1	1	1	?	?
Method 3	0	0	0	?	?

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Alternative approach: evaluate a few pairs first, then decide what to evaluate next

Examples 1-5

Method 1	1	1	1	?	?
Method 2	1	1	1	?	?
~~Method 3~~	0	0	0	?	?

Intuitive illustration only, not theoretically sound

Intuition: how to achieve efficient evaluation?

- Alternative approach: evaluate a few pairs first, then decide what to evaluate next
- We save costs

	Examples 1-5				
Method 1	1	1	1	1	0
Method 2	1	1	1	0	0
~~Method 3~~	0	0	0	?	?

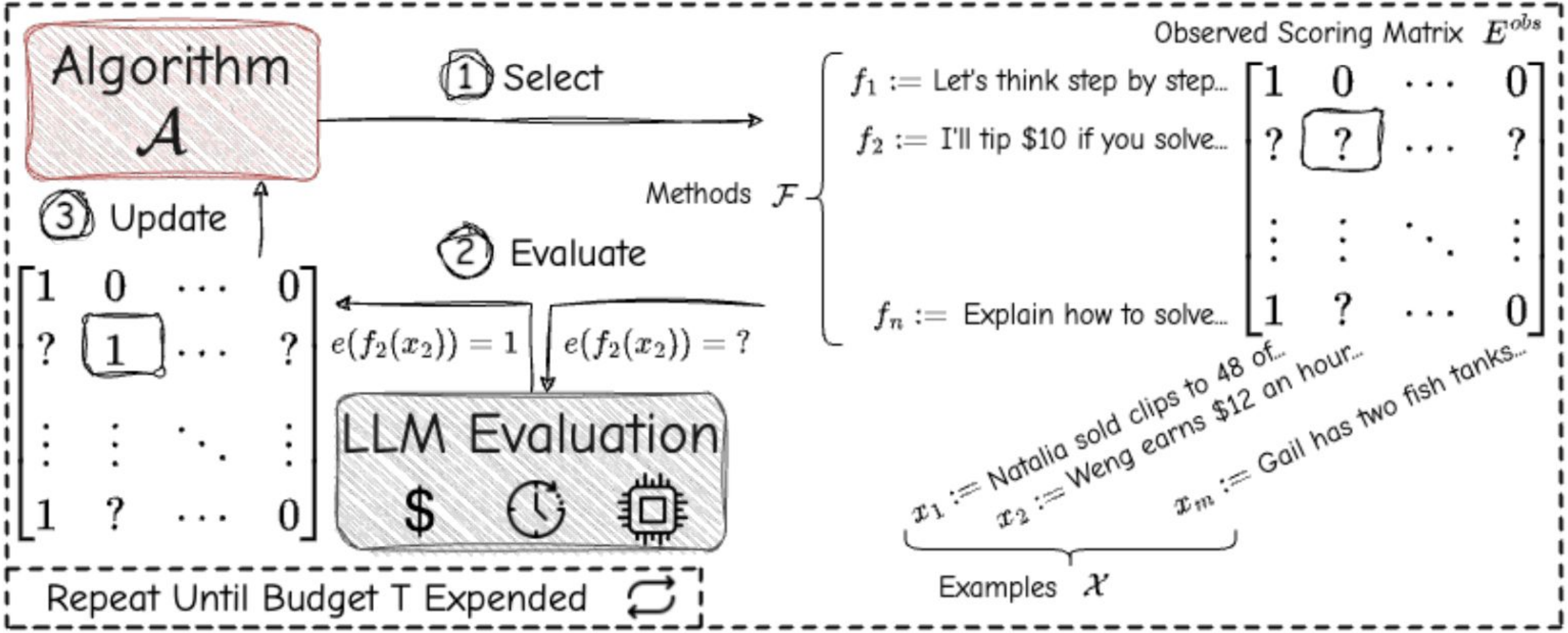
Intuitive illustration only, not theoretically sound

Problem formulation

- Given
  - n methods $\{f_1, \ldots, f_n\}$
  - m examples $\{x_1, \ldots, x_m\}$
  - Scoring function e, $e(f(x))$ -> $[0, 1]$
- Define
  - Ground truth scoring matrix $E \in [0, 1]^{n \times m}$ $E_{ij} := e(f_i(x_j))$
  - Score of a method f_i $\mu_i = \frac{1}{m} \sum_{j=1}^{m} E_{ij}$
- Goal
  - Output the best method $i^* = \arg\max_i \mu_i$

Algorithm: overview

- An adaptive evaluation framework



Algorithm: UCB-E

- Treat each method f_i as an "arm", each pull of f_i receives a random E_{ij}
  - Hence a multi-arm bandit problem
- Compute upper confidence bound (UCB) of each method
- The method with the largest UCB is pulled next

Algorithm 1 UCB-E ($\mathcal{A}_{\mathrm{ue}}(T, \mathcal{F}, \mathcal{X}; a)$)

Input: The evaluation budget T, a set of methods $\mathcal{F}$, a set of examples $\mathcal{X}$, exploration parameter a.

Output: The prediction $\hat{i}^*$ for best method i^*.

The upper confidence bounds $B := \{+\infty\}^n$, the observation matrix $O := \{0\}^{n \times m}$, the observed scoring matrix $E^{\mathrm{obs}} := \{?\}^{n \times m}$.

for $t = 1, \cdots, T$ **do**

    Select: Draw uniformly at random $i \in \arg\max_{k \mid (\sum_{j=1}^{m} O_{kj}) \neq m} B_k$; Draw uniformly at random $j \in \{k \in [m] | O_{ik} = 0\}$.

    Evaluate: Run inference for the method-example pair (f_i, x_j), score the result, and receive $e(f_i(x_j))$; $E_{ij}^{\mathrm{obs}} \leftarrow e(f_i(x_j))$.

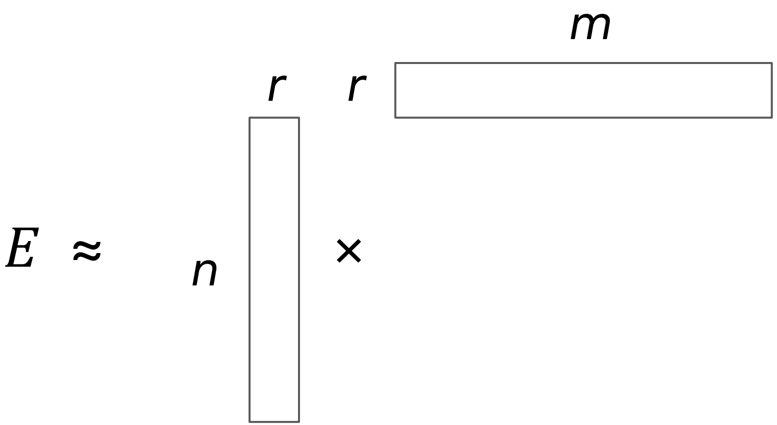
    Update: $O_{i,j} \leftarrow 1$; $B_i \leftarrow \frac{\sum_{j=1}^{m} O_{ij} \cdot E_{ij}^{\mathrm{obs}}}{\sum_{j=1}^{m} O_{ij}} + \sqrt{\frac{a}{\sum_{j=1}^{m} O_{ij}}}$.

end for

Return: $\hat{i}^* = \arg\max_i \frac{\sum_{j=1}^{m} O_{ij} \cdot E_{ij}^{\mathrm{obs}}}{\sum_{j=1}^{m} O_{ij}}$

Can we do better?

- UCB-E has good theoretical guarantee:
  - The chance of selecting the best method converges to 100% by an exponential decay of the number of evaluations
- But does not assume the methods and examples are correlated
- Real-world E are often low-ranked



Algorithm: UCB-E-LRF

- Warm-up for low-rank factorization
- Use ensemble to estimate uncertainty

Algorithm 2 UCB-E-LRF ($\mathcal{A}_{\text{uel}}(T, \mathcal{F}, \mathcal{X}; \mathcal{M}, r, S, T_0, \eta)$)

Input: The evaluation budget T, a set of methods $\mathcal{F}$, a set of examples $\mathcal{X}$, the low-rank factorization model $\mathcal{M}$, the rank r, the ensemble size S, the warm-up budget T_0, the uncertainty scaling η.

Output: The prediction $\hat{i}^*$ for best method i^*.

1: Uniformly draw T_0 method-example pairs from $[n] \times [m]$ and get the observation matrix $O \in \{0,1\}^{n\times m}$ and observed scoring matrix $E^{\text{obs}} \in ([0,1] \cup \{?\})^{n\times m}$ w.r.t. these T_0 evaluations.

2: $\hat{E}, R \leftarrow \mathcal{M}(E^{\text{obs}}, O; r, S)$; $\forall f_i \in \mathcal{F}, B_i \leftarrow \frac{1}{m}\sum_{j=1}^{m}(O_{ij}E_{ij}^{\text{obs}} + (1 - O_{ij})\hat{E}_{ij} + \eta R_{ij})$

3: **for** $t = T_0, \cdots, T$ **do**

4: **Select:** Draw uniformly at random $i \in \arg\max_{k \mid (\sum_{j=1}^{m} O_{kj}) \neq m} B_k$; Draw uniformly at random $j \in \arg\max_{k \mid O_{ik}=0} R_{ik}$.

5: **Evaluate:** Run inference for the method-example pair (f_i, x_j), score the result, and receive $e(f_i(x_j))$; $E_{ij}^{\text{obs}} \leftarrow e(f_i(x_j))$.

6: **Update:** $O_{i,j} \leftarrow 1$; $\hat{E}, R \leftarrow \mathcal{M}(E^{\text{obs}}, O; r, S)$; $\forall f_i \in \mathcal{F}, B_i \leftarrow \frac{1}{m}\sum_{j=1}^{m}(O_{ij}E_{ij}^{\text{obs}} + (1 - O_{ij})\hat{E}_{ij} + \eta R_{ij})$.

7: **end for**

Return: $\hat{i}^* = \arg\max_i \frac{1}{m}\sum_{j=1}^{m}(O_{ij}E_{ij}^{\text{obs}} + (1 - O_{ij})\hat{E}_{ij})$

Datasets

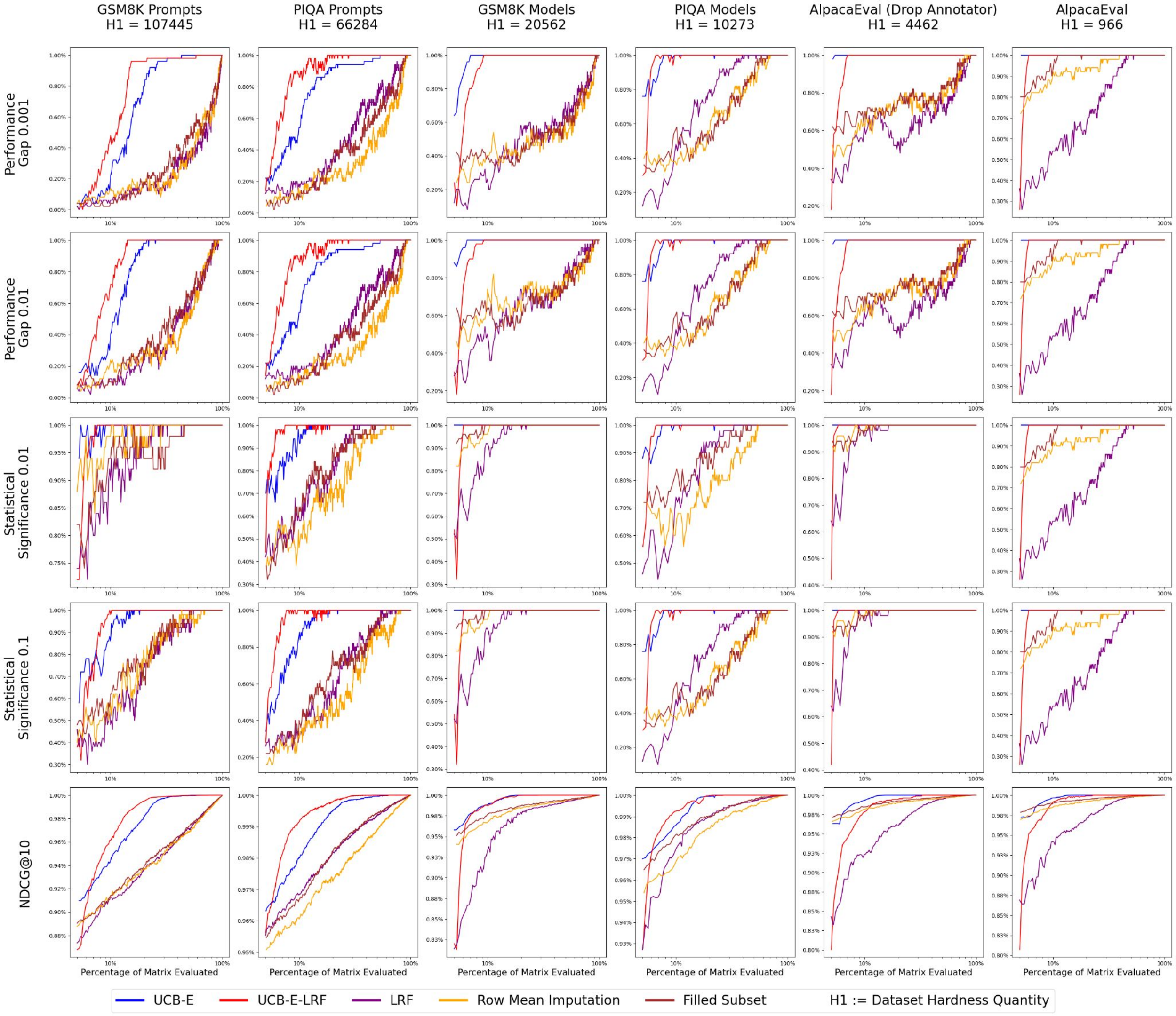
$$H_1 = \sum_{i=1, i \neq i^*}^{n} \frac{1}{(\mu_i - \mu_i^*)^2}$$

- Applications: prompt engineering, model selection, hyperparameter tuning
- Prompt engineering: GPT4-turbo generated prompts
- Model selection: GPT4-turbo as a judge for pairwise comparison
- Hyperparameter tuning: different models + sampling configurations

Dataset Name	Size $n \times m$	Method $\mathcal{F}$	Scoring Function e	H_1
GSM8K Prompts	205×784	Mistral-7B with different prompts	regex match with correct answer	107445
PIQA Prompts	177×1546	Tulu-7B with different prompts	regex match with correct choice	66284
AlpacaEval	154×805	Various LLMs	GPT4-turbo annotator	966
AlpacaEval (Drop Annotator)	153×805	Various LLMs excluding GPT4-turbo	GPT4-turbo annotator	4462
GSM8K Models	122×1000	Various LLMs and sampling configurations	regex match with correct answer	20562
PIQA Models	103×1000	Various LLMs and sampling configurations	regex match with correct choice	10273

Results

- 50 independent runs
- Adaptive algorithms outperform baselines significantly
- Saves as much as 85-95% of costs
- UCB-E-LRF performs better than UCB-E when the dataset is harder



Thank you for your attention!

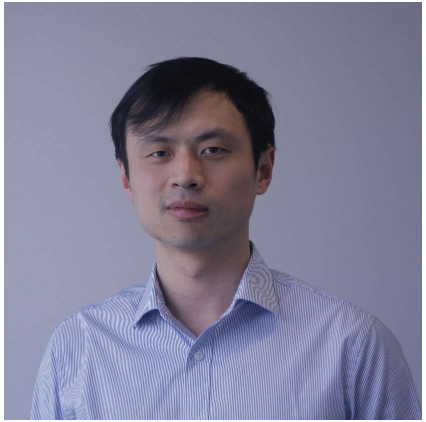
Email: jpzhou@cs.cornell.edu
X: @JinPZhou
Paper: arxiv.org/abs/2407.06172
Code: github.com/kilian-group/banditeval

Christian Belardi

Ruihan Wu

Travis Zhang

Carla Gomes



Wen Sun

Kilian Weinberger