

# On Speeding Up Language Model Evaluation

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

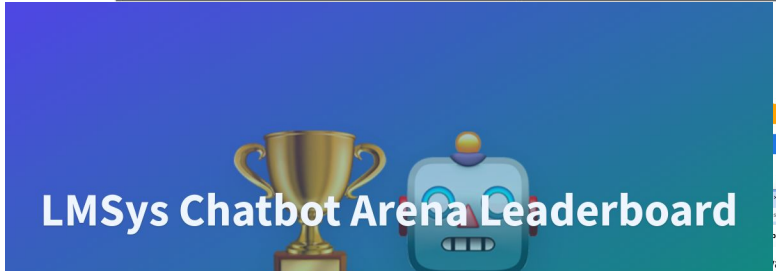
{jz563, ckb73}@cornell.edu, ruw076@ucsd.edu

# LLMs are being evaluated and benchmarked extensively

	Gemini Ultra	Gemini Pro	GPT-4	GPT-3.5	PaLM 2-L	Claude 2
<b>MMLU</b> Multiple-choice questions in 57 subjects (professional & academic) (Hendrycks et al., 2021a)	<b>90.04%</b> CoT@32*	79.13% CoT@8*	87.29% CoT@32 (via API**)	70% 5-shot	78.4% 5-shot	78.5% 5-shot CoT
<b>GSM8K</b> Grade-school math (Cobbe et al., 2021)	<b>94.4%</b> Maj1@32	86.5% Maj1@32	92.0% SFT & 5-shot CoT	57.1% 5-shot	80.0% 5-shot	88.0% 0-shot
<b>MATH</b> Math problems across 5 difficulty levels & 7 subdisciplines (Hendrycks et al., 2021b)	<b>53.2%</b> 4-shot	32.6% 4-shot	52.9% 4-shot (via API**)	34.1% 4-shot (via API**)	34.4% 4-shot	—
<b>BIG-Bench-Hard</b> Subset of hard BIG-Bench tasks written as CoT problems (Srivastava et al., 2022)	<b>83.6%</b> 3-shot	75.0% 3-shot	83.1% 3-shot (via API**)	66.6% 3-shot (via API**)	77.7% 3-shot	—
<b>HumanEval</b> Python coding tasks (Chen et al., 2021)	<b>74.4%</b> 0-shot (PT****)	67.7% 0-shot (PT****)	67.0% 0-shot (reported)	48.1% 0-shot	—	70.0% 0-shot
<b>Natural2Code</b> Python code generation. (New held-out set with no leakage on web)	<b>74.9%</b> 0-shot	69.6% 0-shot	73.9% 0-shot (via API**)	62.3% 0-shot (via API**)	—	—
<b>DROP</b> Reading comprehension & arithmetic. (metric: F1-score) (Dua et al., 2019)	<b>82.4</b> Variable shots	74.1 Variable shots	80.9 3-shot (reported)	64.1 3-shot	82.0 Variable shots	—
<b>HellaSwag</b> (validation set) Common-sense multiple choice questions (Zellers et al., 2019)	87.8% 10-shot	84.7% 10-shot	<b>95.3%</b> 10-shot (reported)	85.5% 10-shot	86.8% 10-shot	—
<b>WMT23</b> Machine translation (met-	<b>74.4</b> 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 <sup>3</sup>	30 / 150 (6th - 12th)	36 / 150 (10th - 19th)	36 / 150 (10th - 19th)
AMC 12 <sup>3</sup>	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

	Gemini Ultra	Gemini Pro	GPT-4	GPT-3.5	PaLM 2-L	Claude 2
						
Model	Arena Elo rating	MT-bench (score)	MMLU	License		
GPT-4	1211	8.99	86.4	Proprietary		
Claude-v1	1169	7.9	75.6	Proprietary		
Claude-Instant-v1	1145	7.85	61.3	Proprietary		
GPT-3.5-turbo	1124	7.94	70	Proprietary		
Vicuna-33B	1096	7.12	59.2	Non-commercial		
Vicuna-13B	1055	6.39	52.1	Non-commercial		
MPT-30B-chat	1049	6.39	50.4	CC-BY-NC-SA-4.0		
Guanaco-33B	1044	6.53	57.6	Non-commercial		
WizardLM-13B	1043	6.35	52.3	Non-commercial		
PaLM-Chat-Bison-001	1019	6.4		Proprietary		
Vicuna-7B	1006	6	47.1	Non-commercial		
Koala-13B	987	5.35	44.7	Non-commercial		
GPT4All-13B-Snoozy	971	5.41	43	Non-commercial		
MPT-7B-Chat	951	5.42	32	CC-BY-NC-SA-4.0		
RMKV-4-Raven-14B	946	3.98	25.6	Apache 2.0		
Alpaca-13B	926	4.53	48.1	Non-commercial		
OpenAssistant-Pythia-12B	919	4.32	27	Apache 2.0		
ChatGLM-6B	904	4.5	36.1	Non-commercial		
FastChat-T5-3B	897	3.04	47.7	Apache 2.0		
StableLM-Tuned-Alpha-7B	867	2.75	24.4	CC-BY-NC-SA-4.0		
RollV-V2-12B	846	3.28	25.7	MIT		
LLaMA-13B	821	2.61	47	Non-commercial		
Claude-2		8.06		Proprietary		
WizardLM-30B		7.01	58.7	Non-commercial		
LLama-2-70B-chat		6.86		LLama 2 Community		



evaluating Reward Bench  
 bench

App
 Files
 Community
 Settings
 Help

# Evaluating Reward Models

## and pitfalls of reward models

(this Paper coming soon)

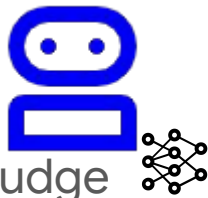
RewardBench - Detailed
 Prior Test Sets
 About
 Dataset Viewer

Seq Classifiers
 ☒
 DPO
 Custom Classifiers
 All Experiments

	Model Type	Average	Chat	Chat Read	Safety	Reasoning	Prior Sets
Seq. Classifier		81.5	96.9	99	89.9	90.3	71.4
DPO		77	97.5	60.8	85.1	89	52.8
seq-7B-Instruct-v0.1		75.8	95	65.2	76.5	92.1	50.3
Seq. Classifier		74.4	98	43.4	88.4	74.6	68.6
DPO		73.9	92.6	62.3	81.7	81.2	52.6
seq-7B-Instruct-v0.1		73.4	92.2	69.5	83.8	76.8	55.5
Seq. Classifier		73.4	93.6	63.2	78	87.6	53.5
DPO		72.9	95.8	56.5	78.4	84.2	49.4
Seq. Classifier		71.3	96.1	55.3	45.8	82	77.2
DPO		70.7	95.2	62.6	54.1	89.6	52.2
DPO		70.4	97.5	54.6	74.3	78.1	47.7
DPO		70.1	86.3	58.2	74	81.3	58.8
DPO		66.4	95.8	51.5	55.1	79.1	51.7
DPO		66.2	62.3	67.3	71.8	87.4	42.3
Seq. Classifier		66.1	89.7	48.9	64.1	76.3	51.7
DPO		66	88	41.3	62.5	73.7	64.6
DPO		65.9	96.6	46.6	68	77.4	48.7
DPO		65.8	57.2	67.4	77.2	85.9	41.2
DPO		65.4	53.6	69.8	75.3	86.4	42.9
Seq. Classifier		65.1	89.5	67.0	63.5	64.4	45.9

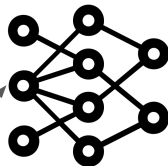
# Evaluation is expensive for benchmarking LLM

Give me a sample 5 day itinerary for a switzerland holiday



Response 1 is better

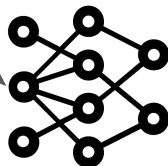
LLM 1



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

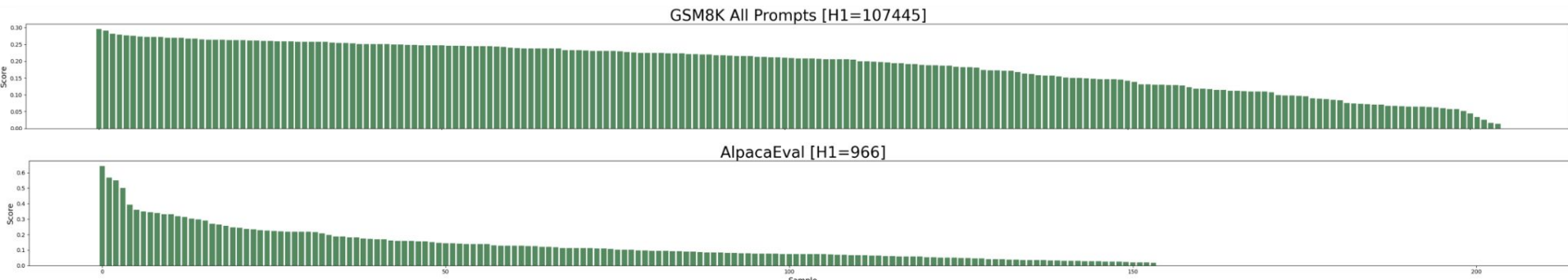
I have been to Switzerland many times and I have never seen a single person who has...

Human / LLM Judge

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhonghao 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

<input type="checkbox"/> 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	?	?
<del>Method 3</del>	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

☐ Method 1

Method 2

~~Method 3~~

Examples 1-5

1	1	1	1	0
1	1	1	0	0
0	0	0	?	?

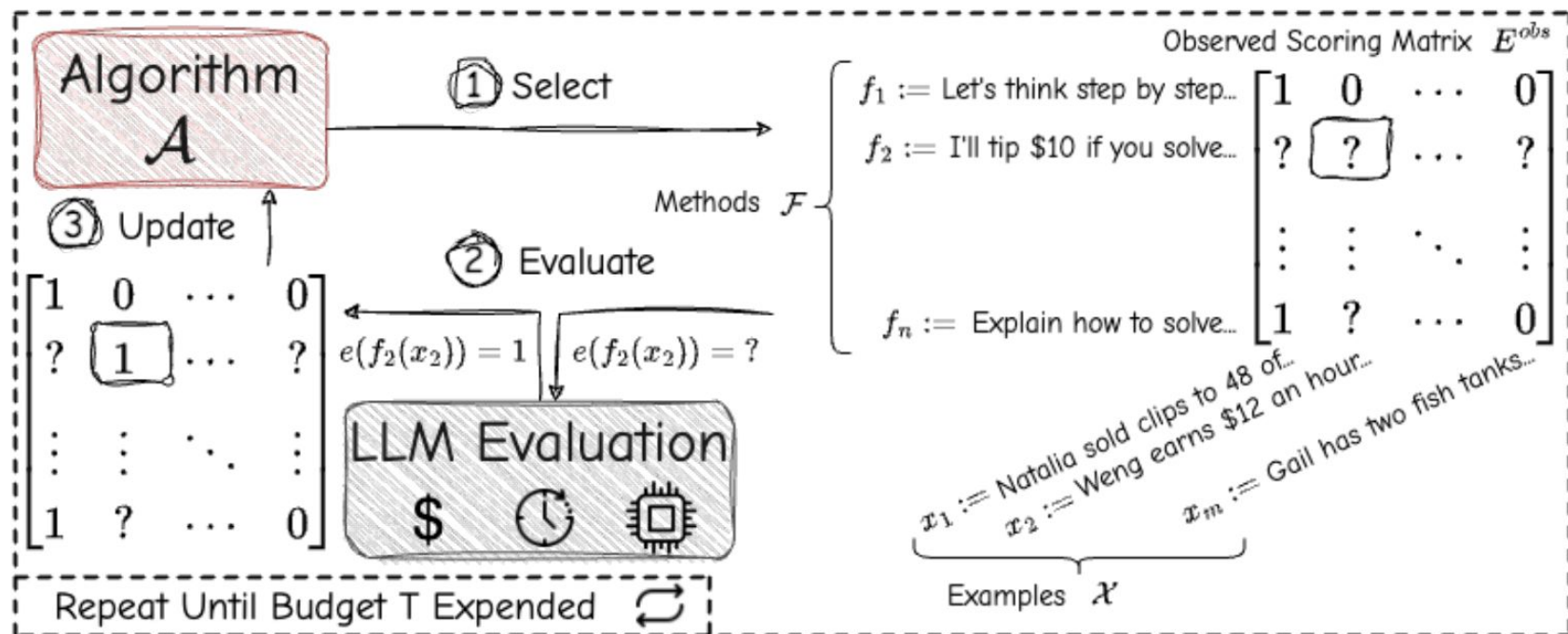
Intuitive illustration only, not theoretically sound

# Problem formulation

- Given
  - $n$  methods  $\{f_1, \dots, f_n\}$
  - $m$  examples  $\{x_1, \dots, x_m\}$
  - Scoring function  $e, e(f(x)) \rightarrow [0, 1]$
- 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 ( $\mathcal{A}_{\text{ue}}(T, \mathcal{F}, \mathcal{X}; a)$ )

---

**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  $i^*$ .

- 1: The upper confidence bounds  $B := \{+\infty\}^n$ , the observation matrix  $O := \{0\}^{n \times m}$ , the observed scoring matrix  $E^{\text{obs}} := \{?\}^{n \times m}$ .
- 2: **for**  $t = 1, \dots, T$  **do**
- 3:   **Select:** Draw uniformly at random  $i \in \arg \max_k |(\sum_{j=1}^m O_{kj}) \neq m| B_k$ ; Draw uniformly at random  $j \in \{k \in [m] | O_{ik} = 0\}$ .
- 4:   **Evaluate:** Run inference for the method-example pair  $(f_i, x_j)$ , score the result, and receive  $e(f_i(x_j))$ ;  $E_{ij}^{\text{obs}} \leftarrow e(f_i(x_j))$ .
- 5:   **Update:**  $O_{i,j} \leftarrow 1$ ;  $B_i \leftarrow \frac{\sum_{j=1}^m O_{ij} \cdot E_{ij}^{\text{obs}}}{\sum_{j=1}^m O_{ij}} + \sqrt{\frac{a}{\sum_{j=1}^m O_{ij}}}$ .
- 6: **end for**

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

---

## 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

Diagram illustrating the approximation  $E \approx n \times r \times m$ . It shows a vertical rectangle labeled  $n$  and a horizontal rectangle labeled  $m$ , with a small square labeled  $r$  at their intersection.



# 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  $\eta$ .

**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**  $t = T_0, \dots, T$  **do**
- 4:   **Select:** Draw uniformly at random  $i \in \arg \max_k |(\sum_{j=1}^m O_{kj}) \neq m| B_k$ ; Draw uniformly at random  $j \in \arg \max_k |O_{ik}=0| R_{ik}$ .
- 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

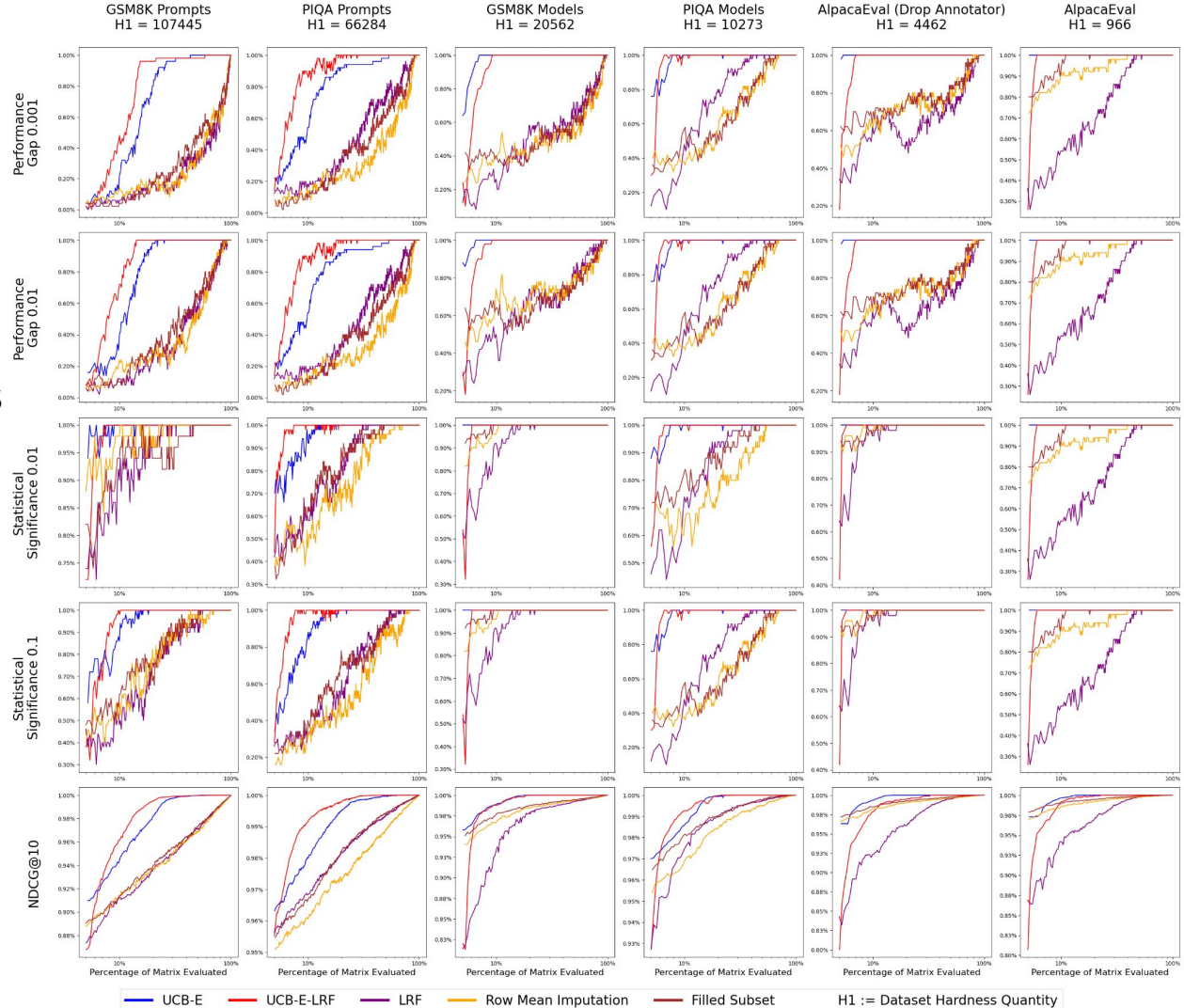
$$H_1 = \sum_{i=1, i \neq i^*}^n \frac{1}{(\mu_i - \mu_{i^*}^*)^2}$$

- 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	$205 \times 784$	Mistral-7B with different prompts	regex match with correct answer	107445
PIQA Prompts	$177 \times 1546$	Tulu-7B with different prompts	regex match with correct choice	66284
AlpacaEval	$154 \times 805$	Various LLMs	GPT4-turbo annotator	966
AlpacaEval (Drop Annotator)	$153 \times 805$	Various LLMs excluding GPT4-turbo	GPT4-turbo annotator	4462
GSM8K Models	$122 \times 1000$	Various LLMs and sampling configurations	regex match with correct answer	20562
PIQA Models	$103 \times 1000$	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](mailto:jpzhou@cs.cornell.edu)

X: @JinPZhou

Paper: [arxiv.org/abs/2407.06172](https://arxiv.org/abs/2407.06172)

Code: [github.com/kilian-group/banditeval](https://github.com/kilian-group/banditeval)



Christian Belardi



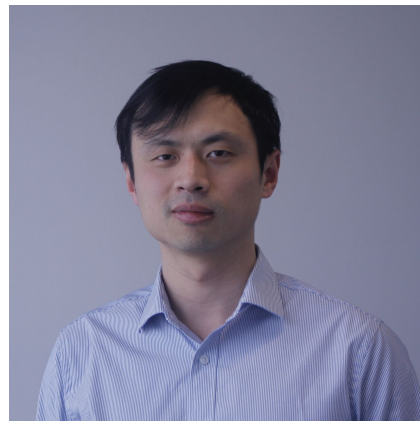
Ruihan Wu



Travis Zhang



Carla Gomes



Wen Sun



Kilian Weinberger