KOREA UNIVERSITY

2024 하계 세미나 발표

어수경

2 Papers

1. When Benchmarks are Targets: Revealing the Sensitivity of Large Language Model Leaderboards (ACL 2024) cite16

2. RouteLLM: Learning to Route LLMs with Preference Data (Arxiv, 2024.06)

When Benchmarks are Targets: Revealing the Sensitivity of Large Language Model Leaderboards

Norah A. Alzahrani[*], **Hisham Abdullah Alyahya**[*], **Yazeed Alnumay**, **Sultan Alrashed**, **Shaykhah Z. Alsubaie**, **Yousef Almushayqih**, **Faisal Abdulrahman Mirza**, **Nouf M. Alotaibi** **Nora Al-Twairesh**, **Areeb Alowisheq**, **M Saiful Bari**, **Haidar Khan**[*†]

National Center for AI (NCAI), Saudi Data and AI Authority (SDAIA)
Riyadh, Saudi Arabia

Introduction

- LLM은 benchmark 또는 leaderboard를 바탕으로 능력의 우수성을 결정
- 하지만 이런 benchmark들이 LLM의 능력을 제대로 평가하려면 개선되어야 할 여지가 여전히 많이 남아있음

- 평가를 위한 Multiple choice questions(MCQ)의 장단점
  - 장점: 자동화, 정량화 가능한 수단
  - 단점:
    (1) 성능 평가를 위한 안정화된 수단은 아님 → small perturbation만으로도 rank가 크게 바뀌기도 함
    (2) benchmark에 모델이 overfit(spurious correlation, pattern recognition, optimization for specific question formats, etc.) → real-world applicability의 부족

→ Requirements of abilities that mirrors the complexity of real-world use

→ 본 연구에서는 MCQ benchmark에 대해 perturbation experiment 진행, model ranking과의 연관성을 밝힘

LLM Evaluation with MCQs

- MCQ tests: MMLU, ARC, CSQA, etc.
- Minor changes (extra spaces, additional structural phrases) -> 성능 변화에 영향
- Pezeshkpour and Hruschka (2023) -> Changes in order can change model's prediction

- Variations in three categories:
  (1) Answer choice format and ordering: ordering, formatting 변경
  (2) Prompt and scoring modifications: prompt text 및 scoring scheme 변경
  (3) In-context knowledge manipulation: prompt 또는 few-shot example에 관련/관련되지 않은 정보 주입

LLM Evaluation with MCQs

- Answer choice format and ordering

[Random choice order]

- Swapping choices -> 모든 questions에 대해 정해진대로 swap
- Randomly assigning new positions (모든 choices는 본래의 위치에 있지 않도록 함)

[Biased choice order]

- Choice가 특정 position에 bias되어 나타나는지 확인
- Zero-shot: correct answer를 모든 test set에 대해 동일한 위치로 설정
- Few-shot: example 내의 correct answers가 특정 position에 내재적으로 bias되어 있을 때 answer choice에 편향을 가지는지 확인

LLM Evaluation with MCQs

- Answer choice format and ordering

[Answer choice symbols]

- Answer choice symbols (A,B,C,D) -> alternative, less common tokens로 변경
- Symbols, relative ordering으로부터 particular position bias 분리 목적

(1) Common tokens (language independent): ["$", "&", "#", "@"]

(2) Rare tokens (implicit relative order X) : ["œ", "§", "Ze (Cyrillic)", "ü"]

LLM Evaluation with MCQs

- Prompt and scoring modifications
  - →LLMs exhibit high sensitivity to variations in prompt formatting (Sanh et al., 2021; Mishra et al., 2022)
  - →모델에 따라 특정 prompt에 우세할 수 있음

- 3 major scoring methods of MCQs
  - (1) Symbol scoring: answer choice symbol에 대한 likelihood score 기반 선택
  - (2) Hybrid scoring: answer choice content 에 대한 likelihood score (normalized by length)
  - (3) Cloze scoring:Q 제공, answer choice에 대한 maximum normalized likelihood score

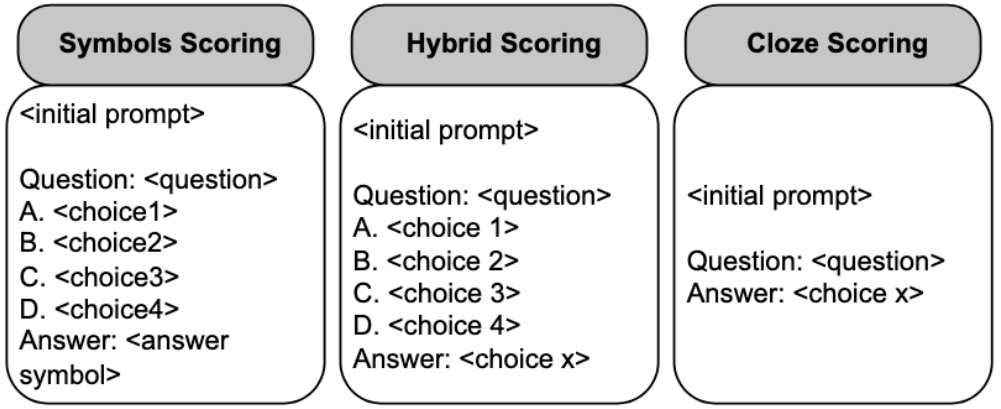


Figure 3: Answer choice scoring methods for LLMs. The symbols and hybrid scoring methods are most similar, sharing identical prompts. Cloze scoring does not reflect a "true" MCQ style, as the model is not shown all the options. However, due to its prevalence we compare it to the other methods as a baseline.

LLM Evaluation with MCQs

- In-context knowledge manipulation
  - Entire spectrum of knowledge injection in the few-shot examples → model and benchmark robustness 측정 목적

[Correct answer provided] target question + correct answer (as an example)

[Incorrect answer provided] target question + incorrect answer (as an example)

[Trivial examples] 모델이 대답할 수 있을 것으로 알려진 간단한 질문으로 대체 (example -> formatting에 대한 정보만 제공)

[Out-of-domain examples] 다른 도메인의 예제 제공

Experiments

- Dataset: MMLU + ARC challenge(일부)
- Metric:
  - Delta-acc: change in accuracy
  - Recall standard deviation(Rstd): 각 answer choice에 대한 recall의 표준 편차 계산 → 특정 answer choice에 대한 모델 편향을 측정. 즉, 모델이 올바른 답변 선택에 대해 특정 위치의 선호 정도를 정량화

Results and Analysis

1. MCQ benchmarks are not robust to perturbations

- Perturbation
  → dramatic shifts in the order of models

- Yi-6b : 3rd → 6/7th
- Mistral-7b , Llama2-7b: rank 2-3 차이

→ 특정 benchmark style에 대해 overfit 가능성 (학습데이터를 모르므로 검증은 불가)

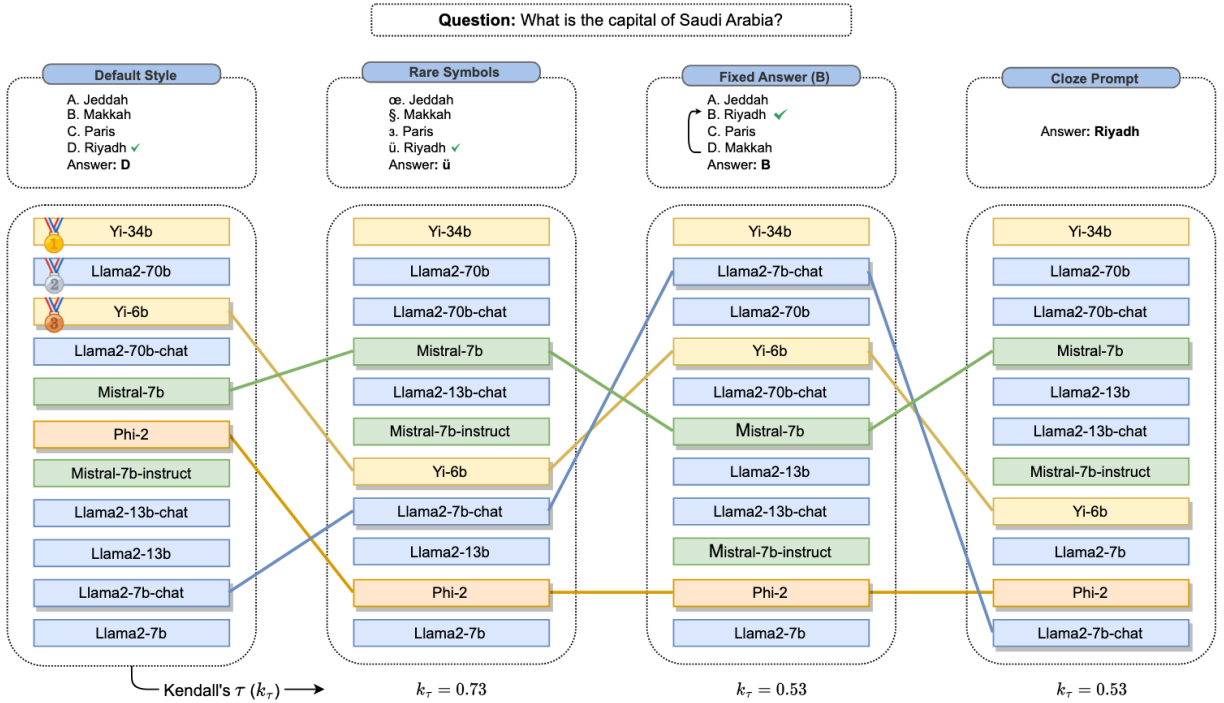


Figure 1: Minor perturbations cause major ranking shifts on MMLU (Hendrycks et al., 2020). Models can move up or down up to eight positions on the leaderboard under small changes to the evaluation format. Columns (from left): 1) Original ranking given by MMLU using answer choice symbol scoring (a common default). 2) Ranking under an altered prompt for the same questions, where answer choice symbols are replaced with a set of rare symbols. 3) Setting where the correct answer choice is fixed to a certain position (in this case, B). 4) Using the cloze method for scoring answer choices. Under each new ranking, we report Kendall's τ (Kendall, 1938) with respect to the original ranking (lower k_τ indicates more disagreement between rankings)

Results and Analysis

1. MCQ benchmarks are not robust to perturbations [정답을 특정 위치로 고정]

Model	Baseline	A	B	C	D
phi-2	54.47	52.31 (-2.16)	56.53 (+2.07)	56.30 (+1.83)	50.19 (-4.28)
Yi-6b	61.12	62.53 (+1.41)	64.44 (+3.32)	58.59 (-2.53)	63.13 (+2.02)
Mistral-7b	59.56	52.19 (-7.38)	60.98 (+1.42)	63.84 (+4.27)	60.43 (+0.86)
Mistral-7b-Instruct	53.48	49.77 (-3.71)	54.67 (+1.18)	49.99 (-3.49)	57.74 (+4.26)
Llama2-7b	41.81	66.36 (+24.55)	30.40 (-11.42)	36.28 (-5.53)	23.37 (-18.44)
Llama2-7b-chat	46.37	30.84 (-15.53)	69.41 (+23.04)	50.05 (+3.68)	28.23 (-18.14)
Llama2-13b	52.08	35.82 (-16.26)	57.24 (+5.16)	68.65 (+16.57)	44.08 (-8.00)
Llama2-13b-chat	53.12	36.73 (-16.39)	56.72 (+3.60)	71.81 (+18.69)	42.63 (-10.49)
Yi-34b	73.38	66.16 (-7.22)	75.22 (+1.84)	78.07 (+4.69)	73.88 (+0.50)
Llama2-70b	65.44	56.47 (-8.97)	67.38 (+1.95)	69.92 (+4.48)	66.47 (+1.03)
Llama2-70b-chat	61.11	41.78 (-19.34)	62.24 (+1.13)	75.07 (+13.96)	57.71 (-3.41)
k_τ	-	0.455	0.527	0.527	0.855

Placing the correct answer at each possible position (zero-shot)

→ Selection bias는 모든 LLM에서 나타남

Results and Analysis

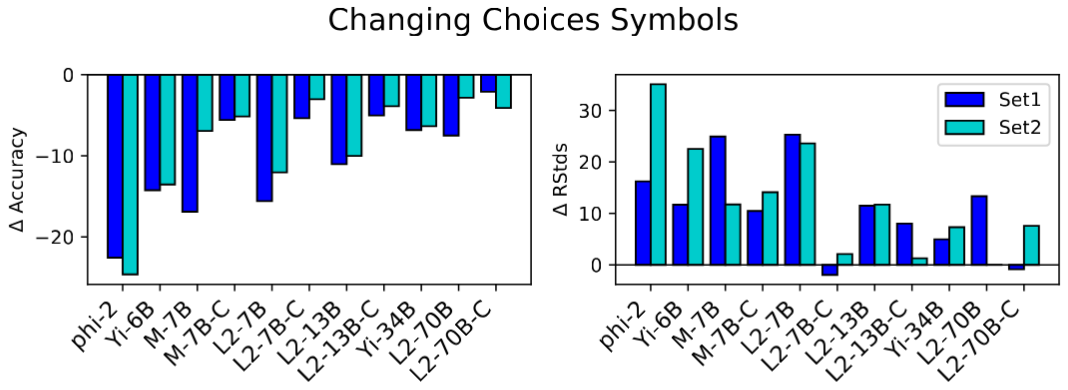
Delta-acc: change in accuracy
Recall standard deviation(Rstd): 특정 answer choice에 대한 모델 편향 측정
→ Std ↑: answer choice 편향 ↑

1. MCQ benchmarks are not robust to perturbations

Model	Rank	Acc (ΔAcc)	RStd (ΔRStd)
phi-2	(7→7)	34.6 (-3)	14.2 (7.4)
Yi-6b	(3→9)	33.0 (-8.3)	11.9 (1.8)
Mistral-7b	(4→3)	40.0 (1.0)	9.8 (0.7)
Mistral-7b-Instruct	(8→8)	33.3 (-1.7)	16.7 (3.5)
Llama2-7b	(11→11)	24.3 (-5.0)	13.2 (-0.4)
Llama2-7b-chat	(9→10)	28.6 (-3.7)	27.7 (7.9)
Llama2-13b	(6→6)	37.0 (0.7)	22.7 (5.7)
Llama2-13b-chat	(9→5)	37.6 (6.0)	26.7 (0.0)
Yi-34b	(1→1)	45.0 (-5.0)	9.2 (-2.3)
Llama2-70b	(2→2)	40.3 (-1.7)	9.07 (-5.5)
Llama2-70b-chat	(5→4)	37.6 (0.3)	13.4 (-6.2)

[symbol 고정, text 셔플]
Shuffling of the order of answer choices (zero-shot)

→ 5 out of 11 models change in ranking



["$", "&", "#", "@"]
["œ", "§", "Ze (Cyrillic)", "ü"]

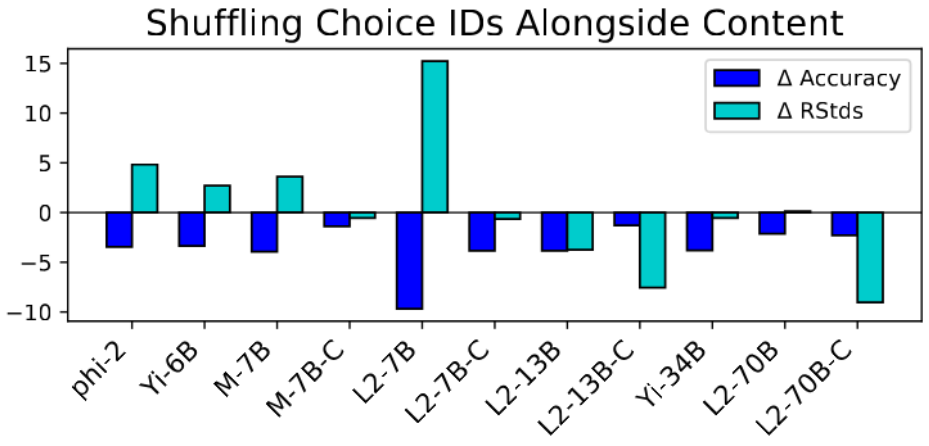
[symbol swapping, text 고정]
Swapping answer choice symbols with two different sets of symbols (zero-shot) – 단순 기호로만 변경
→ 성능은 모두 떨어짐
→ 새로운 symbol에서 더 큰 편향

→ MCQ Benchmark는 perturbation을 적게 주더라도 성능이 크게 변화

Results and Analysis

2. Revisiting selection bias: token bias vs. position bias

- Selection bias는 모든 LLM의 setup에서 발견
- 다음 스텝으로, positional bias와 token bias를 disentangle해보고자 함

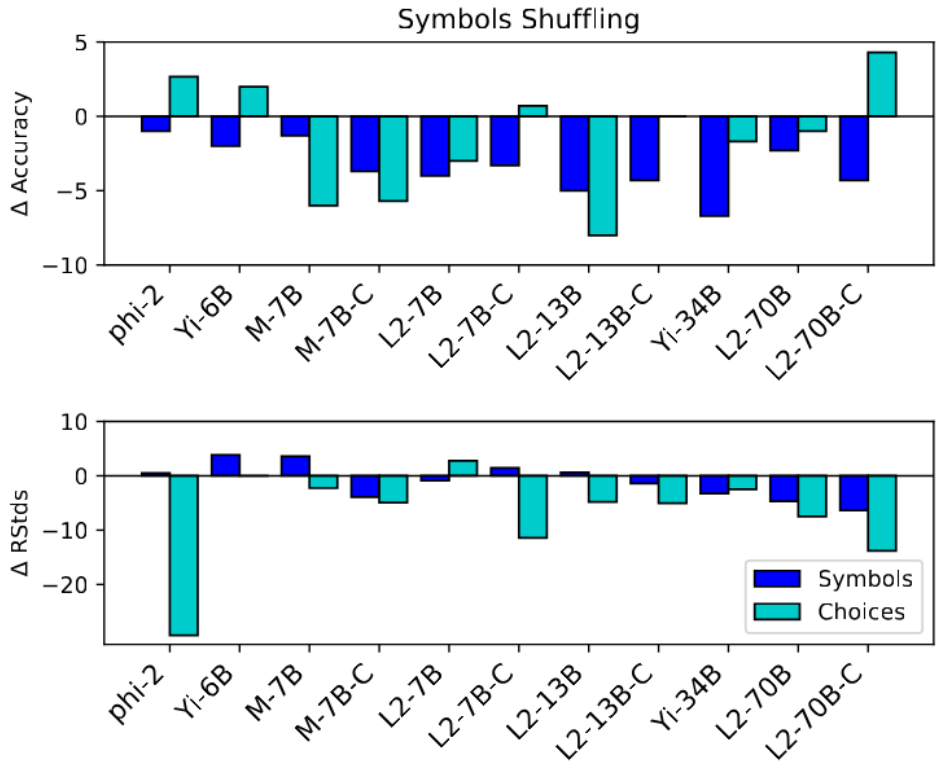


비교 실험 - [choice id, text 모두 셔플]
Choice id와 text 모두 shuffle: 그래도 bias됨
→ 단순히 choices 전체를 shuffle한다고 해서 positional bias가 사라지지 않음

→ Symbol을 바꿔보자 (토큰 편향을 위치 편향으로부터 분리)
(A/B/C/D) -> rare symbol without an implicit relative ordering

Results and Analysis

2. Revisiting selection bias: token bias vs. position bias



파랑: symbol 변경 후 symbol만 shuffle
시안: symbol 변경 후 symbol 고정, answer choice 내 text 변경

→ Delta 측정을 위한 비교값: symbol 단순히 1:1로 바꾼 결과
→ Symbol 단순 변경한다고 bias가 줄어들지 않음
→ Symbol 에 shuffle까지 하더라도 bias는 unpredictable하게 변화

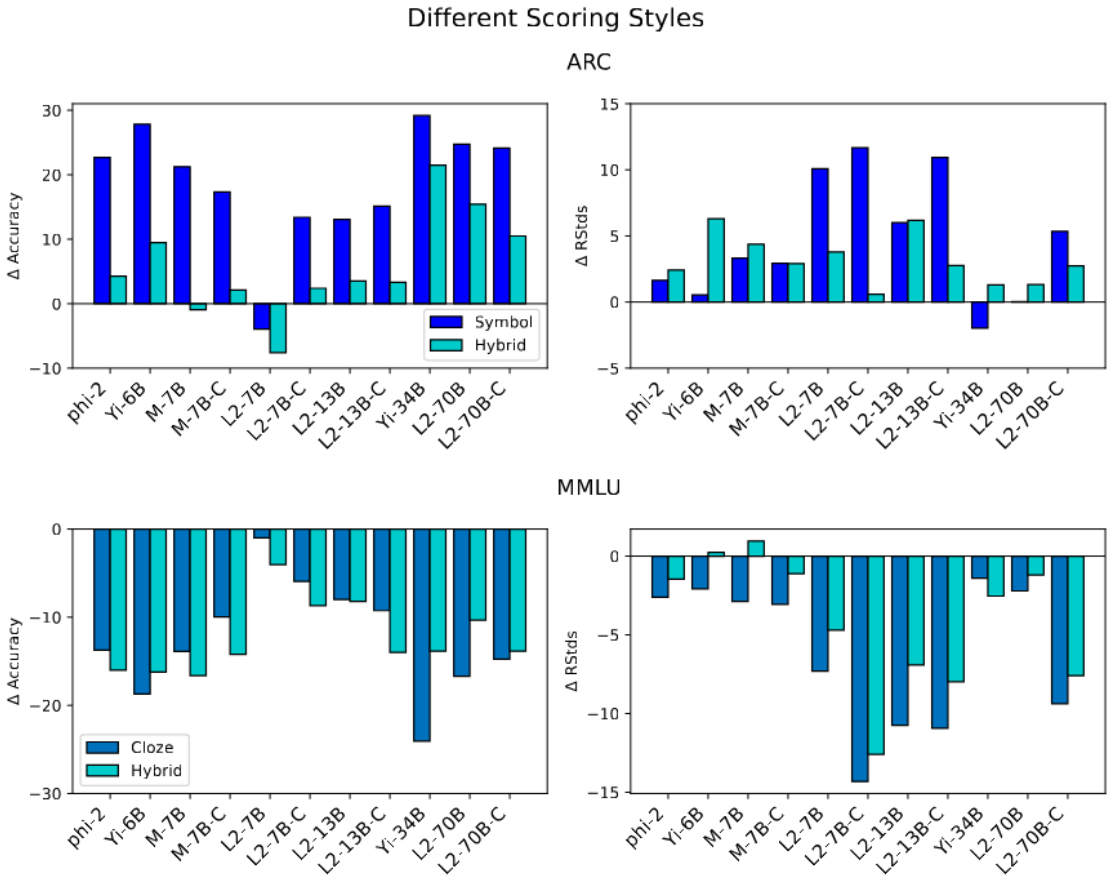
(1) LLM은 choice ID를 나타내는 symbol에 대해 언제나 편향이 있음
(2) symbol을 shuffle한다 하더라도 bias는 예측 불가능한 형태로 나타남

Results and Analysis

3. Another source of bias: scoring bias

- Accuracy : symbol > hybrid > cloze
- Bias: symbol > hybrid > cloze

- Symbol scoring: 성능은 전체적으로 높게 나옴, but selection bias는 커질 수 있음
- Cloze scoring: bias는 줄일 수 있다 하더라도 성능이 대부분 낮음
- Hybrid scoring: 모든 answer choice가 나오고 이 answer를 생성하도록 한 다음 normalized score 뽑았었음 -> balance between upper two
  → 모델 랭킹 시 bias를 줄이기 위해 이를 사용하는 걸 추천



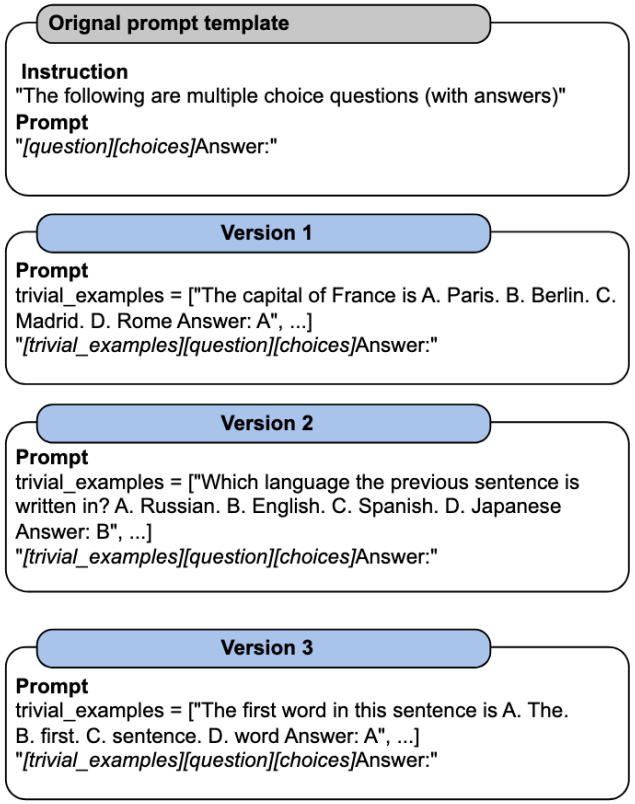
* Arc -> baseline cloze, mmlu -> baseline symbol

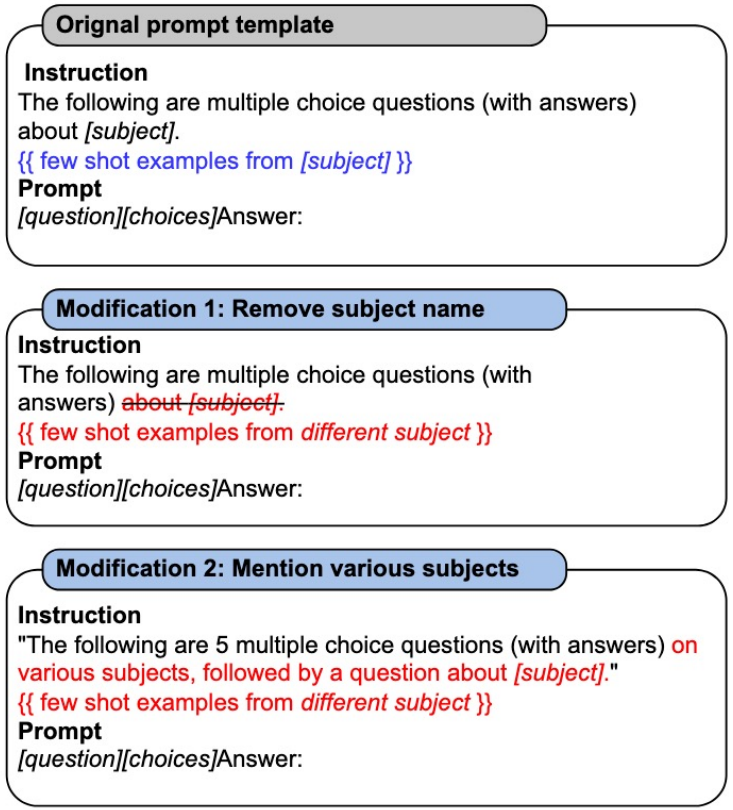
Results and Analysis

4. Minor few-shot and prompt changes have little effect on benchmark rankings

- 관련 있는 example → 관련 없는 example / 너무 당연한 example / 관련 없는 도메인 example 로 각각 변경
- Zero-shot에 비해 bias 조금 줄어들었음

→few-shot example을 조금 다르게 주는 것이 도움이 될 수 있음





Results and Analysis

5. LLMs readily reference knowledge provided in-context (even if it is misleading)

- LLM에게 in-context example을 줄 때 모든 answer position을 하나로 고정
- 즉, 5 example의 정답을 모두 하나의 symbol로 고정

	5-shot Baseline	A	B	C	D
phi-2	56.77	36.67 (-20.11)	41.33 (-15.44)	40.67 (-16.11)	41.67 (-15.11)
Yi-6B	63.22	36.67 (-26.56)	36.33 (-26.89)	37.67 (-25.56)	39.33 (-23.89)
Mistral-7B	62.36	34.67 (-27.70)	41.33 (-21.03)	43.00 (-19.36)	40.33 (-22.03)
Llama-2-7b	45.88	22.00 (-23.88)	31.00 (-14.88)	30.67 (-15.22)	34.33 (-11.55)

평가 결과 모든 모델에서 성능 하락
→ 모델은 작은 **패턴**들에도 집중을 하고 있음
→ few-shot example을 어떻게 주느냐 역시 중요

Conclusion

- LLM의 벤치마크/리더보드는 아주 적은 perturbation에도 크게 반응 → 전체 모델 랭킹에도 큰 영향을 줌
- 특히나 position bias, token bias, scoring bias 등 다양한 bias들로 인해 모델은 selection bias가 생길 수 있음
- 뿐만 아니라 few-shot example을 줄 때에도 answer의 position이 현재 answer 선택에 있어 큰 영향을 줄 수 있음
- 따라서 이런 변화에 robust한 방법론을 고려할 필요가 있음

- 제안 방향:
  - 사실상 symbol, 선지의 위치 등을 변화시킨다고 해서 bias를 줄이는 것은 어려움
  - 따라서 벤치마크 평가 시 hybrid scoring(answer text 생성 후 normalized score평가) 진행
  - Zero-shot보다는 few-shot 이 조금 더 robust할 것 (대신 선지에 일관성을 두지 않아야 하며, trivial, irrelevant, ood example을 조금씩 섞어보는 편이 좋아보임)

RouteLLM: Learning to Route LLMs with Preference Data

Isaac Ong[†]
UC Berkeley
isaacong@berkeley.edu

Amjad Almahairi[†]
Anyscale
anm@anyscale.com

Vincent Wu
UC Berkeley
cveinnt@berkeley.edu

Wei-Lin Chiang
UC Berkeley
weichiang@berkeley.edu

Tianhao Wu
UC Berkeley
thw@berkeley.edu

Joseph E. Gonzalez
UC Berkeley
jegonzal@berkeley.edu

M Waleed Kadous
Canva
waleed@canva.com

Ion Stoica
UC Berkeley & Anyscale
istoica@berkeley.edu

Introduction

- 최근 LLMs는 다양한 task에 대해 우수한 성능을 보이고 있음
- 모든 LLM들이 동일하게 만들어지진 않음
  → size, cost 등에 따라 다양 (one billion to hundreds of billions of params)
  → 학습한 데이터에 따라 다름: 모델간 강점/약점/특정 능력들이 각각 다름
- 일반적으로 큰 모델은 성능 좋음, high cost / 작은 모델은 성능은 비교적 낮지만 low cost)

- real-world에서 practical deployment 할 때 딜레마가 생김 → LLM 라우팅 제안
  - 이상적 시나리오:
    - 비교적 쉬운 질의: 작은 사이즈 모델 활용을 통한 비용 절감
    - 복잡한 질의: 큰 사이즈의 모델을 활용해 성능 향상

LLM Routing

- Problem Formulation

- A set of N different LLM models: $M = \{M_1, \dots, M_N\}$, $M_i: Q \rightarrow A$(maps query to answer)
- Routing function: R
- Question : q, answer: $a = M_{R(q)}(q)$
- Preference data가 있다고 가정 $D_{pref} = \{(q, l_{i,j}) | q \in Q, i, j \in N, l_{i,j} \in L\}$
  *$l_{i,j}$: M_i, M_j의 퀄리티를 비교한 결과 label
- Reward modeling: post-LLM generation의 quality 평가
- Routing: response 생성 이전 적절한 모델을 선택
  - 2 classes of models를 라우팅: (1) strong models(M_{strong})- GPT4, (2) weak model (M_{weak})- Mistral7B
  - Binary routing function $R^{\alpha}_{bin}: Q \rightarrow \{0,1\}$

LLM Routing

- Problem Formulation
- 2 components of R_{bin}^{α}

(1) **Win prediction model**: strong model의 winning probability 예측

$$P_\theta\left(win_{M_{strong}} \middle| q\right) = \max_\theta \sum_{(q,l_{i,j}) \in D_{pref}} \log P_\theta(l_{i,j} | q)$$

(2) **Cost threshold**: $\alpha \in [0,1]$, winning probability → routing decision 변환

$$R_{bin}^{\alpha}(q) = \begin{cases} 0(i.e. M_{weak}) \; if \; P\left(win_{M_j} \middle| q\right) < \alpha, \\ 1\left(i.e., M_{strong}\right) \; otherwise \end{cases}$$

-> alpha 값이 클수록 stricter cost constraint, reducing expenses, compromising quality

따라서, $a = M_{R(q)}(q)$ → $a = M_{R_{bin}^{\alpha}(q)}(q)$ (router 결과에 따라 모델 선정 및 query 제공 후 응답 생성)

LLM Routing

- **Metrics**: capturing trade-off between cost-quality
- 2 compounded metrics

(1) Cost efficiency: *percentage of calls to the strong model*

$$c(R_{bin}^{\alpha}) = \frac{1}{|Q|}\sum_{q \in Q} I\{ R_{bin}^{\alpha}(q) = 1\}$$

(2) Quality: *average response quality* on an evaluation set Q

$r(R_{bin}^{\alpha}) = \frac{1}{|Q|}\sum_{q \in Q} \delta(M_{R_{bin}^{\alpha}(q)}(q))$, $\delta(M_{R_{bin}^{\alpha}(q)}(q))$: router response의 수치 스코어

→ predefined metric의 결과 (e.g. MMLU 성능), numerical label(e.g. 1-5, 5-10) 가능

LLM Routing

- Router's performance 정량화: strong/weak 모델 간 성능 격차를 기준으로 라우터 성능 정량화

- Performance gap recovered(PGR): R_{bin}^{α}의 전반적인 성능 향상을 정의하는 지표 (단순 성능 측정)

$$\mathrm{PGR}(R_{bin}^{\alpha}) = \frac{r(R_{bin}^{\alpha}) - r(M_{weak})}{r(M_{strong}) - r(M_{weak})}$$

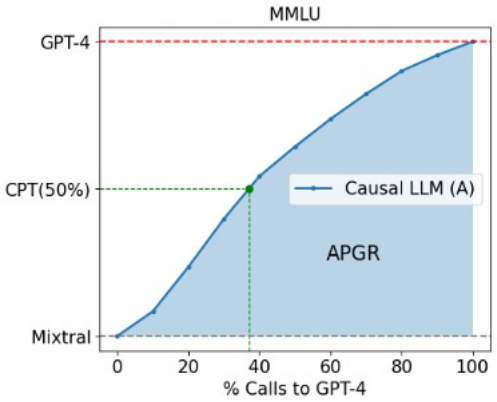
  → 하지만 이 식은 quality-cost trade-off를 capture하지는 못함(모든 query를 strong model로 보내면 PGR=1)

- Average performance gap recovered (APGR): 서로 다른 성능 제약에서도 성능 차이가 얼마나 recover되는지를 측정

$$\mathrm{APGR}(R_{bin}) = \int_0^1 R_{bin}^{\alpha}\, d(c(R_{bin}^{\alpha}))$$

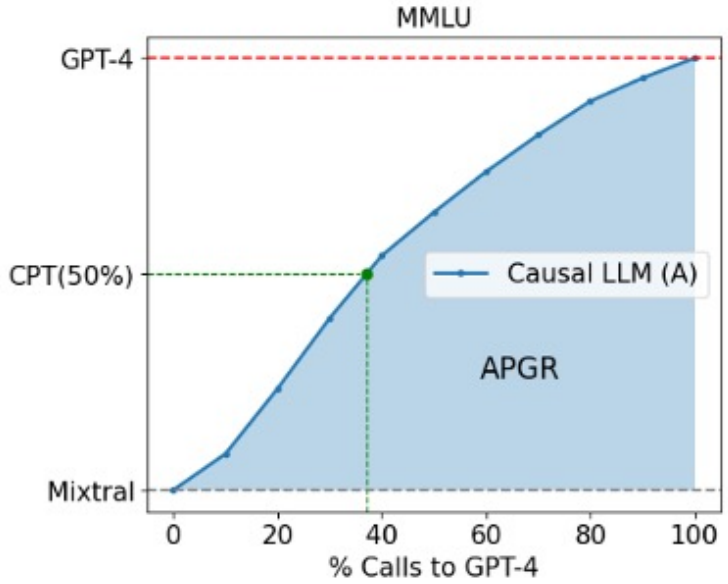
  →Discrete하게 쪼갬 (strong model을 부를 확률-[0%,100%]를 10등분($\{c_i\}_{i\in 10}$, 각 c_i마다 threshold α_i 선택)

$$\mathrm{APGR}(R_{bin}) \approx \frac{1}{10}\sum_{i=1}^{10} \mathrm{PGR}(R_{bin}^{\alpha_i})$$



LLM Routing

- Real-world application에서는 “특정 level의 성능을 달성하기 위한 cost”를 정량화해야 함
  → second metric call-performance threshold(CPT)
- Desired router performance (PGR $x\%$)
  → CPT는 이 PGR $x\%$를 달성하기 위해 최소한으로 요구되는 strong model의 percentage를 의미



→ 50% PGR이라는 성능을 달성하기 위해 최소한으로 요구되는 strong model percentage : 약 37%

Methodology

Preference data

- Router training을 위한 dataset
- Online chatbot arena platform 활용: 사용자들은 2 anonymous models의 아웃풋에 대해 winning model or tie를 선택
- Resulting dataset $D_{arena} = \{(q, a_i, a_j, l_{i,j}) | q \in Q, a_i, a_j \in A, l_{i,j} \in L\}$

  * User queries, answers from 2 models M_i, M_j, human judgement 기반 label

Data augmentation

- Golden-labeled datasets: $D_{gold} = \{(q, a, l_g) | q \in Q, a \in A, l_g \in \mathbb{R}\}$ → model answer에 대해 자동으로 gold label 계산 (MMLU benchmark 등 활용) -> gold label 응답을 win으로 간주
- LLM-judge-labeled datasets: open-ended purpose chat domains에 대해 LLM judge를 활용하여 label 계산. GPT-4 활용해서 두 모델 비교 후 labeling

Methodology - Routing Approaches

- Preference data를 활용한 win prediction model 학습 $P_\theta\left(win_{M_{strong}} \middle| q\right)$
- $e = (q, M_w, M_L)$, M_w, M_l→ winning model, losing model

1. Similarity-weighted (SW) ranking: 학습 없이 inference에서만 작동. Bradley-Terry (BT) model 기반(항목 쌍간 비교를 통한 상대적 강도 측정)
    - $w_i = \gamma^{1+S(q,\hat{q})}$: 사용자 query q가 주어졌을 때 각 query q_i에 대해 q와의 similarity 기반 weight 구함(γ=10)
    - ϵ , ϵ_i (query embedding)

$$S(q, q_i) = \frac{\epsilon \cdot \epsilon_i}{||\epsilon||\,||\epsilon_i|| \cdot \max_{1 \le s \le |D_{pref}|} \frac{\epsilon_i \cdot \epsilon_s}{||\epsilon_i||\,||\epsilon_s||}}$$

→ (User query와, train set 내의 쿼리 i간 유사도) x (1/쿼리 i와 가장 유사한 embedding의 유사도 점수)
→ User query와 쿼리 i의 임베딩: 0.5 x 1/0.5 = 1 (*쿼리 i와 가장 높은 유사도를 지니는 쿼리의 유사도 점수: 0.5)
→ User query와 쿼리 j의 임베딩: 0.2 x 1/0.6 = 0.333.. (*쿼리 j와 가장 높은 유사도를 지니는 쿼리의 유사도 점수: 0.6)
→ 비교할 쿼리와 가장 유사도가 높은 점수를 먼저 파악하고, 이를 분모로 해서 현재 query의 최종 weight를 결정

Methodology - Routing Approaches

- Preference data를 활용한 win prediction model 학습 $P_\theta\left(win_{M_{strong}} \middle| q\right)$
- 앞서 구한 w_i 를 바탕으로 similarity-weighted ranking 진행

$$argmin_\xi \sum_{i=1}^{|D_{pref}|} \left[w_i \cdot \ell\left(l_i, \frac{1}{1+e^{\xi_{w_i}-\xi_{l_i}}} \right) \right]$$

- 각 데이터마다 (similarity weight x binary cross-entropy loss) summation → 각 데이터마다 유사도 값이 높으면 그만큼 해당 데이터에 대한 loss 값이 중요해짐.
- 특히나 weight가 높은 데이터 index에서 이런 loss 값이 낮은 것이 중요

→ 이렇게 얻은 값은 다음 <u>win probability</u> 값을 예측 한 것으로도 볼 수 있음:

$$\mathrm{P}\left(win_{M_w} \middle| q\right) = \frac{1}{1+e^{\xi_w - \xi_l}} \text{ (Bradley-terry model)}$$

<u>(정리) strong model의 winning probability를 예측하는 모델 (학습X), logistic function 기반이며 user query와 학습 데이터 queries를 비교→ user query와의 유사도를 바탕으로 중요도를 주면서 argmin이 되는 확률 값 포인트를 찾겠다</u>

Methodology

2. Matrix factorization

- Score $s(M_w, q)$: query q에 대해 모델 M_w의 응답 퀄리티
  →Model M_w 가 M_l 보다 q에 대해 더 잘한다 : $s(M_w, q) > s(M_l, q)$
- 이를 바탕으로 win probability를 모델링

$$P\left(win_{M_w} \middle| q\right) = \sigma(s(M_w, q) - s(M_l, q)),$$ → 0-1사이의 값 출력

- Scoring function s

$$s(M, q) = w_2^T \left(v_m \odot \left(W_1^T v_q + b\right)\right), \mathrm{W}_1 \in \mathbb{R}^{d_q \times d_m}, b \in \mathbb{R}^{d_m}$$

  * s : bilinear function of model and query
  * v_m: 모델 M의 identity embedding → d_m dim vector
  * v_q: query q에 대한 임베딩 → d_q dim vector
  * $w_2 \in \mathbb{R}^{d_m}$: 최종 scalar 값을 얻기 위한 linear regression layer

Methodology

3. BERT classifier

- BERT-base architecture를 활용한 text classification

$$P_\theta\left(win_{M_w} \middle| q\right) = \sigma(Wh_{CLS} + b)$$

4. Causal LLM classifier

- Llama3 8B architecture 활용

- Experiments :
  - Training data → 80K chatbot arena
  - Evaluation benchmarks → MMLU, MT Bench, GSM8K
  - Routers → similarity-weighted ranking router: OpenAI embedding3-small
    - Strong model: gpt-4-1106-preview, weak model: Mixtral 8x7B

Results

Figure 1: Routing performance/cost trade-off between GPT-4 and Mixtral-8x7B. *(left)* We demonstrate several routers that outperform the random baseline on OOD eval GSM8K. *(center)* We demonstrate improvement in router performance through data augmentation, denoted by (A), on MT Bench. *(right)* We display the main metrics we consider: call-performance threshold (CPT, denoted in green) and average performance gain recovered (APGR, denoted by the blue shaded region).

- Routing 성능 – cost 간 trade-off를 나타낸 figure (GPT-4를 100%로 부를수록 성능은 좋아질 것)
- [left] random baseline보다 성능 향상
- [center] DA를 통해 성능을 더 향상시킬 수 있음 (낮은 cost로 높은 성능)
- [right] call-performance threshold(CPT), average performance gain recovered (APGR) → 측정할 것

Results

- MT Bench results

Training data	Method	$CPT(50\%)$	$CPT(80\%)$	$APGR$	
	Random (95% CI)	49.03(±4)%	78.08(±3)%	0.500(±0.02)	(+0%)
$\mathcal{D}_{\text{arena}}$	BERT	78.09%	87.64%	0.391	(-21.8%)
	Causal LLM	28.82%	77.53%	0.573	(+14.6%)
	Matrix Factorization	**25.32%**	74.26%	0.580	(+16%)
	SW Ranking	37.85%	**58.99%**	**0.610**	(+22.1%)
$\mathcal{D}_{\text{arena}} + \mathcal{D}_{\text{judge}}$	BERT	19.58%	34.02%	0.751	(+50.2%)
	Causal LLM	31.50%	48.75%	0.679	(+35.8%)
	Matrix Factorization	**13.40%**	**31.31%**	**0.802**	(+60.4%)
	SW Ranking	23.21%	36.04%	0.759	(+51.8%)

- Matrix factorization, similarity-weighted ranking에서 우수한 성능
- Augmentation 이후에는 BERT 성능 역시 좋아짐
- Augmentation 이후 best model: matrix factorization
  → GPT-4를 random에 비해 50% 이상 덜 호출-성능 80%까지 달성

Results

- MMLU results

Training data	Method	$CPT(50\%)$	$CPT(80\%)$	$APGR$	
	Random (95% CI)	50.07(±0)%	79.93(±0)%	0.500(±0)	(+0%)
$\mathcal{D}_{\text{arena}}$	BERT	49.43%	77.80%	0.502	(+0.5%)
	Causal LLM	48.88%	77.93%	0.499	(-0.2%)
	Matrix Factorization	**45.00%**	**76.86%**	**0.524**	(+4.9%)
	SW Ranking	55.82%	80.25%	0.473	(-5.4%)
$\mathcal{D}_{\text{arena}} + \mathcal{D}_{\text{gold}}$	BERT	41.30%	72.20%	0.572	(+14.4%)
	Causal LLM	35.49%	**70.31%**	0.600	(+19.9%)
	Matrix Factorization	35.46%	71.40%	0.597	(+19.5%)
	SW Ranking	**35.40%**	71.55%	**0.603**	(+20.7%)

- Random보다 성능이 낮거나 크게 향상되지 않음
- Augmentation 후에는 성능이 눈에 띄게 향상됨

Results

- GSM8K results

Training data	Method	$CPT(50\%)$	$CPT(80\%)$	$APGR$	
	Random (95% CI)	50.00(±2)%	80.08(±1)%	0.497(±0.01)	(+0%)
$\mathcal{D}_{\text{arena}}$	BERT	58.78%	83.84%	0.438	(-11.8%)
	Causal LLM	56.09%	83.56%	0.461	(-7.3%)
	Matrix Factorization	**53.59%**	85.24%	0.4746	(-4.5%)
	SW Ranking	54.43%	**82.11%**	**0.4753**	(-4.3%)
$\mathcal{D}_{\text{arena}} + \mathcal{D}_{\text{judge}}$	BERT	44.76%	79.09%	0.531	(+6.9%)
	Causal LLM	**33.64%**	**63.26%**	**0.622**	(+25.3%)
	Matrix Factorization	38.82%	72.62%	0.565	(+13.8%)
	SW Ranking	41.21%	72.20%	0.568	(+14.3%)

- MMLU 결과와 유사한 경향
- Causal LLM에서 가장 우수한 결과 (17%정도 GPT-4를 덜 호출, 50%의 성능 확보)

Results

- Quantifying dataset and benchmark similarity
- 성능이 비교적 낮은 이유: evaluation data 와 training data간 distribution이 달랐기 때문
- benchmark-dataset similarity score를 계산해봤음

	Arena	Arena augmented with with $\mathcal{D}_{\text{judge}}$	Arena augmented with $\mathcal{D}_{\text{gold}}$
MT Bench	0.6078	0.6525	-
MMLU	0.4823	-	0.5678
GSM8K	0.4926	0.5335	-

- MT Bench와는 유사도가 Arena에서 상대적으로 MMLU, GSM8K보다 높음 → 더 우수한 효과를 MT Bench에서 보았을 것

Results

- Generalizing to other model pairs
- 다른 model pairs도 실험: strong → Claude 3 Opus, weak → Llama3 8B
- 새롭게 retraining을 하지 않고 단순 strong, weak 모델만 교체

Train Set	Method	$CPT(50\%)$	$CPT(80\%)$	$APGR$	
	Random (95% CI)	47.23(±5)%	77.08(±5)%	0.494(±0.03)	(+0%)
$\mathcal{D}_{\text{arena}}$	BERT	60.61%	88.39%	0.475	(-3.9%)
	Causal LLM	31.96%	**59.83%**	**0.645**	(+30.5%)
	Matrix Factorization	**24.84%**	85.73%	0.605	(+22.5%)
	SW Ranking	33.54%	80.31%	0.553	(+12%)
$\mathcal{D}_{\text{arena}} + \mathcal{D}_{\text{judge}}$	BERT	36.28%	50.83%	0.618	(+25.2%)
	Causal LLM	40.46%	55.83%	0.625	(+26.5%)
	Matrix Factorization	**30.48%**	**41.81%**	**0.703**	(+42.2%)
	SW Ranking	31.67%	48.39%	0.667	(+35%)

- 모델이 바뀌어도 추가 training 없이 좋은 라우팅 성능을 보임
- Router 모델이 strong, weak 모델을 구분할 수 있는 common characteristics를 가지는 것으로 봄

Results

- Cost analysis
- Cost saving에 대한 정도 정량화

	CPT 50%	CPT 80%
MT Bench	3.66x (at 95% GPT-4 quality)	2.49x
MMLU	1.41x (at 92% GPT-4 quality)	1.14x
GSM8K	1.49x (at 87% GPT-4 quality)	1.27x

Conclusion

- Strong model, weak model 간 cost-quality trade-off를 고려한 라우팅
- 라우팅 모델 총 4개
  - 주로 LLM은 데이터가 늘어날수록, 성능을 80%로 맞출수록 우수
  - 학습데이터와 유사한 evaluation benchmark인 MT Bench에서는 matrix factorization, similarity-weighted ranking이 더 우수 (이런 모델이 일반화 능력은 살짝 떨어진다고 볼 수 있음)

- Strong/weak의 binary가 아닌 multiple models로 확장하는 편이 좋아보임
  → pair-wise 비교 또는 multi-class classification 등으로 수행가능

Thank you!

Q&A