2024 하계 세미나

심규호

Natural Language
Processing
& Artificial Intelligence

고려대학교
KOREA UNIVERSITY

Is DPO Superior to PPO for LLM Alignment? A Comprehensive Study

Shusheng Xu [1] Wei Fu [1] Jiaxuan Gao [1] Wenjie Ye [2] Weilin Liu [2]
Zhiyu Mei [1] Guangju Wang [2] Chao Yu [*1] Yi Wu [*123]

Toward Informal Language Processing: Knowledge of Slang in Large Language Models

Zhewei Sun[1], Qian Hu[2], Rahul Gupta[2], Richard Zemel[2,3], Yang Xu[1]

[1]University of Toronto, [2]Amazon Alexa AI, [3]Columbia University
{zheweisun, yangxu}@cs.toronto.edu, {huqia, gupra}@amazon.com
zemel@cs.columbia.edu

Is DPO Superior to PPO for LLM Alignment? A Comprehensive Study

Shusheng Xu [1] Wei Fu [1] Jiaxuan Gao [1] Wenjie Ye [2] Weilin Liu [2]
Zhiyu Mei [1] Guangju Wang [2] Chao Yu [*1] Yi Wu [*1 2 3]

2024 ICML

심규호

Natural Language Processing & Artificial Intelligence

고려대학교 KOREA UNIVERSITY

0. Abstract

강화학습(RLHF)은 크게 두 가지 부류, **reward-based**와 **reward-free**가 있는데, **PPO**와 **DPO**가 각 부류의 대표적인 방법론 중 하나임.

- **Reward-Based** - e.g. PPO (Proximal Policy Optimization)
- **Reward-Free** - e.g. DPO (Direct Performance Optimization)

Q1. DPO는 PPO보다 우월한가? Is DPO truly superior to PPO?
Q2. 왜 PPO는 아카데믹 벤치마크에서 활약을 하지 못하는가? Why does PPO perform poorly on Academic Benchmarks?

→

1. **DPO는 근본적인 한계가 존재함**
    a. **DPO**의 알고리즘 성질들에 대한 이론적 **&** 경험적인 연구
2. **PPO의 LLM Fine-tuning 성능 최적화를 위한 Critical Factor 소개**
    a. **PPO**에 대한 종합적인 평가

⇒ ***PPO는 다른 alignment methods들을 모두 능가 & challenging code competitions에서 SOTA 기록***

1. Introduction

거대언어모델 **LLM**의 방대한 능력들을 실제에 적용하기 위한 **human preference**에 **align**하는 연구 활발

⇒ **Fine-Tuning**의 전형적인 **steps**

1. **SFT Supervised Fine-tuning**를 통한 베이스모델 구축
2. **RLHF Reinforcement Learning from Human Feedback**를 통한 모델의 성능 향상

   a. Reward-Based

      1. **PPO**

   b. Reward-Free

      2. **DPO, RRHF, PRO**

1. Introduction

Academic Benchmark에서 DPO가 PPO보다 더 활약을 하는 경향

⇒

1. **RLHF domain에서 DPO가 PPO보다 우월한가?**
2. **RLHF 벤치마크에서 PPO의 성능이 발전될 수 있는가?**

⇒

이론적(Theoretical) & 경험적(Empirical) 분석 진행

1. Introduction

이론적 & 경험적 분석

1) DPO는 out-of-distribution responses 분포 밖의 답변을 통해 biased solution을 내는 경향
   - DPO performance이 모델 output & preference dataset 사이의 dist'N shift의 큰 영향을 받는 점

Ablation 연구

2) PPO의 중요한 알고리즘 관련 요소 탐구
   - Advantage normalization, large batch size, exponential moving average

다양한 실험을 통한 findings 검증

3) PPO는 꾸준히 DPO를 능가하는 성능을 보임
   - Dialogue generation task, Code Completion task

2. Related Work

RLHF: 인간의 선호도를 반영하기 위해 인간 피드백 기반 강화학습

- **Reward-based**
  - **Reward Model을 Preference data에 훈련**
  - **학습된 Reward Model은 PPO와 같은 RL 강화학습 알고리즘에 reward signal을 보내줌**
- **Reward-free**
  - **LLM을 preference data에 바로 학습시키거나 human preference를 distill하기 위한 data ranking**
  - **가장 좋은 퍼포먼스를 보인 DPO**

Academic Benchmark에서 DPO ←→ Chat-GPT, Claude와 같이 성공적인 application들은 PPO 기반

3. Preliminary

a) Language Model 언어 모델

- 언어 모델을 하나의 **Policy** 취급 $\pi_\theta(\mathbf{y} \mid \mathbf{x})$

  - **user instruction X**를 입력 받았을 때 **text response y**를 받게끔 가이드

$$\pi_\theta(\mathbf{y} \mid \mathbf{x}) = \prod_t \pi_\theta(y_t \mid \mathbf{x}, \mathbf{y}_{<t}), \qquad (1)$$

  - **prompt x**를 받았을때 **auto-regressive**한 방식으로 답변 **y**를 생성

3. Preliminary

b) SFT Supervised Fine-Tuning

- 모델 **alignment**의 초기 단계

c) RLHF Reinforcement Learning from Human Preference

- **SFT**에서 더 나아가 **alignment**하는 단계
- 아래의 **obiective**를 극대화하는 방식

$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]. \quad (2)$$

- **r: reward function**
- **πref: Kullback-Leibler Divergence**와 함께 **πθ**를 **regularize**하는데 사용되는 **reference model**
- **β: regularization** 정규화의 **degree**를 조절하는 상수

3. Preliminary

$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]. \quad (2)$$

c-i) PPO Proximal Policy Optimization

1. **Reward model Training**
   a. r(x, y)가 주어지지 않았다면 reward model $r_\phi \in R$ 으로 r를 추정해야함
      i. human-annotated data $\mathcal{D} = \{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)\}$로 학습
      ii. y_w가 y_l보다 나은 답변일 확률

$$\mathbb{P}_\phi(\mathbf{y}_w \succ \mathbf{y}_l \mid \mathbf{x}) = \frac{\exp\left(r_\phi(\mathbf{x},\mathbf{y}_w)\right)}{\exp\left(r_\phi(\mathbf{x},\mathbf{y}_w)\right) + \exp\left(r_\phi(\mathbf{x},\mathbf{y}_l)\right)}$$
$$= \sigma\left(r_\phi(\mathbf{x},\mathbf{y}_w) - r_\phi(\mathbf{x},\mathbf{y}_l)\right). \quad (3)$$

   b. r_ϕ 훈련은 negative-log-likelihood의 최소화 과정을 통해 진행

$$\mathcal{L}_R(r_\phi) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}}\left[\log\sigma(r_\phi(\mathbf{x},\mathbf{y}_w) - r_\phi(\mathbf{x},\mathbf{y}_l))\right] \quad (4)$$

3. Preliminary

c-i) PPO Proximal Policy Optimization

2. 모델/policy πθ 훈련: RLHF의 Objective을 극대화(maximize) 진행함
   - 이때 $J_r(\pi_\theta) \rightarrow J_{r_\phi}(\pi_\theta)$

$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]. \tag{2}$$

3. Preliminary

$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]. \quad (2)$$

c-ii) DPO Direct Policy Optimization

1. **Reward Model의 학습 과정 없이 곧바로 모델/policy πθ를 preference data에 훈련시킴**

a.

$$\pi^*(\mathbf{y}\mid\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})\exp\left(\frac{1}{\beta}r(\mathbf{x},\mathbf{y})\right), \quad (5)$$

b.

$$r_\phi(\mathbf{x},\mathbf{y}) = \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})} + C(\mathbf{x}). \quad (6)$$

c.

$$\mathcal{L}_{\text{DPO}}(\pi_\theta) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}} \quad (7)$$

$$\left[\log\sigma\left(\beta\left(\log\frac{\pi_\theta(\mathbf{y}_w\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w\mid\mathbf{x})} - \log\frac{\pi_\theta(\mathbf{y}_l\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l\mid\mathbf{x})}\right)\right)\right].$$

3. Preliminary

$$\mathcal{L}_R(r_\phi) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}}\left[\log\sigma(r_\phi(\mathbf{x},\mathbf{y}_w) - r_\phi(\mathbf{x},\mathbf{y}_l))\right] \tag{4}$$

$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]. \tag{2}$$

$$\mathcal{L}_{\text{DPO}}(\pi_\theta) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}} \tag{7}$$
$$\left[\log\sigma\left(\beta\left(\log\frac{\pi_\theta(\mathbf{y}_w\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w\mid\mathbf{x})} - \log\frac{\pi_\theta(\mathbf{y}_l\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l\mid\mathbf{x})}\right)\right)\right].$$

4. DPO의 한계

"DPO는 PPO보다 뛰어나지 않을 수도 있다"

1) 이론적 분석 → **DPO training Objective**과 관련된 이슈-**DPO**의 한계점
2) **Synthetic Scenario**을 통한 실험적 검증 → **DPO**는 **out-of-dist'N data**에 취약함
3) 실제 **preference dataset**에 실험 → **DPO performance**를 향상시키는 방법

4. DPO의 한계- 이론적 분석

- PPO가 학습된 Reward-Model의 잠재적 결함들의 영향을 받아 잘못된 output으로 이끄는 경향이 있다는 연구 존재
- DPO는 Reward-Modeling을 안하지만, 비슷한 일반화 문제가 존재함
- 아래의 제시된 정리는 왜 DPO가 PPO보다 열약한지 설명해주는 논문의 핵심적인 역할을 함

"PPO로 찾아진 policy들은 모두 DPO로 생성된 policy들의 진부분집합이다"

→ 모든 PPO로 찾아진 solution은 DPO objective를 minimize한다

→ *Reward Model의 결함들이 이끌어낸 모든 PPO의 solution들은 DPO를 통해서도 도출할 수 있다*

→ 더불어, DPO는 추가적으로 out-of-distribution data를 통해 잘못 도출된 solution들을 찾아서 human preference와 정렬이 잘 된 reference policy로부터 크게 멀어질 수 있다.

Theorem 4.1. *Given a ground-truth reward r and a preference dataset $\mathcal{D}$, let Π_{PPO} be the class of policies induced by training reward model r_ϕ over $\mathcal{D}$ and running PPO to optimize $J_{r_\phi}(\theta)$. Let Π_{DPO} be the class of policies induced by minimizing DPO objective Eq. (7). We have the following conclusion: Π_{PPO} is a proper subset of Π_{DPO}.*

4. DPO의 한계- 이론적 분석

Action	$\mathbf{y}_1$	$\mathbf{y}_2$	$\mathbf{y}_3$
π_{ref}	0.5	0.5	0
D_{pref}	$\{(\mathbf{y}_w = \mathbf{y}_1, \mathbf{y}_l = \mathbf{y}_2)\}$		
π_{DPO}	0.1	0.0	0.9
π_{PPO}	1	0	0

π_{DPO} 를 생성하지만 π_{ref} 의 제한 때문에 **PPO**는 같은 **policy**를 생성하지 못할것이다.

Table 1. A state-less counter-example with three actions when DPO can minimize the loss but produce an unexpected policy. PPO will not produce π_{DPO} because π_{ref} enforces the probability of outputting $\mathbf{y}_3$ is zero.

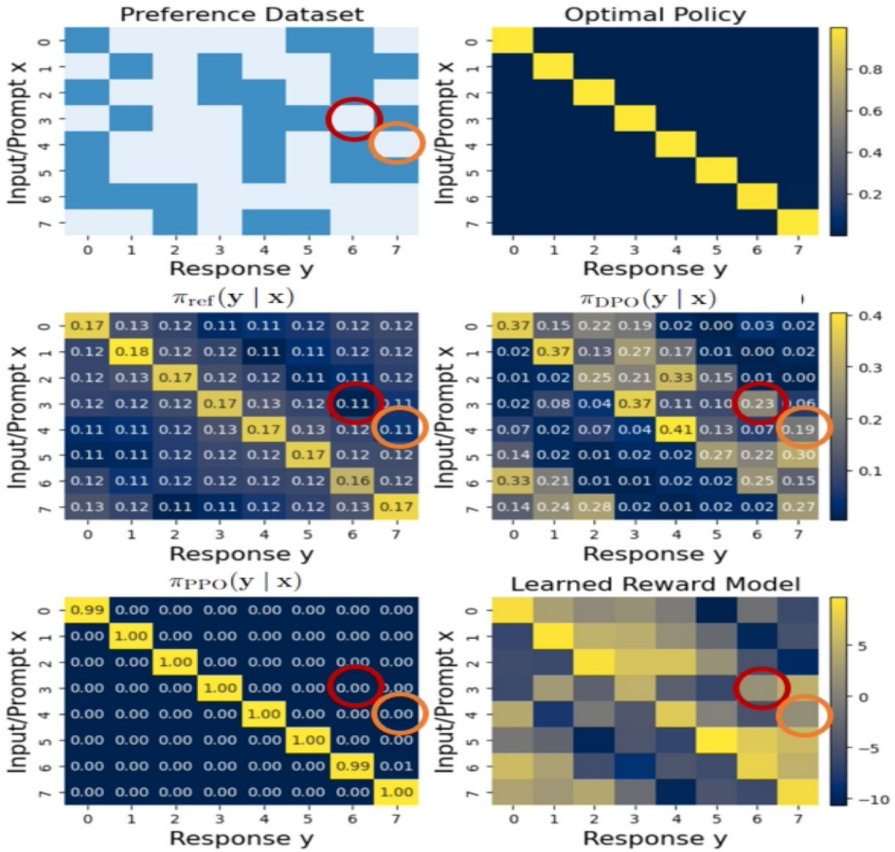
4. DPO의 한계- 이론적 분석

Reward Misspecification Issue; 강화학습에서 모델이 원하는 방향으로 **align**이 되지 않는 이슈

- 핵심 원인은 **Preference dataset**의 너무 좁은 **Dist'N** 범위
- **PPO**의 경우, 학습된 **Reward Model**이 **Out-of-Dist'N** 샘플들에게 큰 점수를 주는 것이 문제가 됨
- **DPO**는 **Reward Model training**을 피하지만, 다른 방식으로 **OOD** 샘플에 **Misspecification** 문제가 발생함
  - **DPO**는 **unseen response**를 편애하는 편향된 분포를 생성할 수 있음
  - 이와 달리, **PPO**는 **preference dataset**의 분포 이외에도 **prompt-only data & 그에 대해 생성된 답변**을 활용할 수 있음

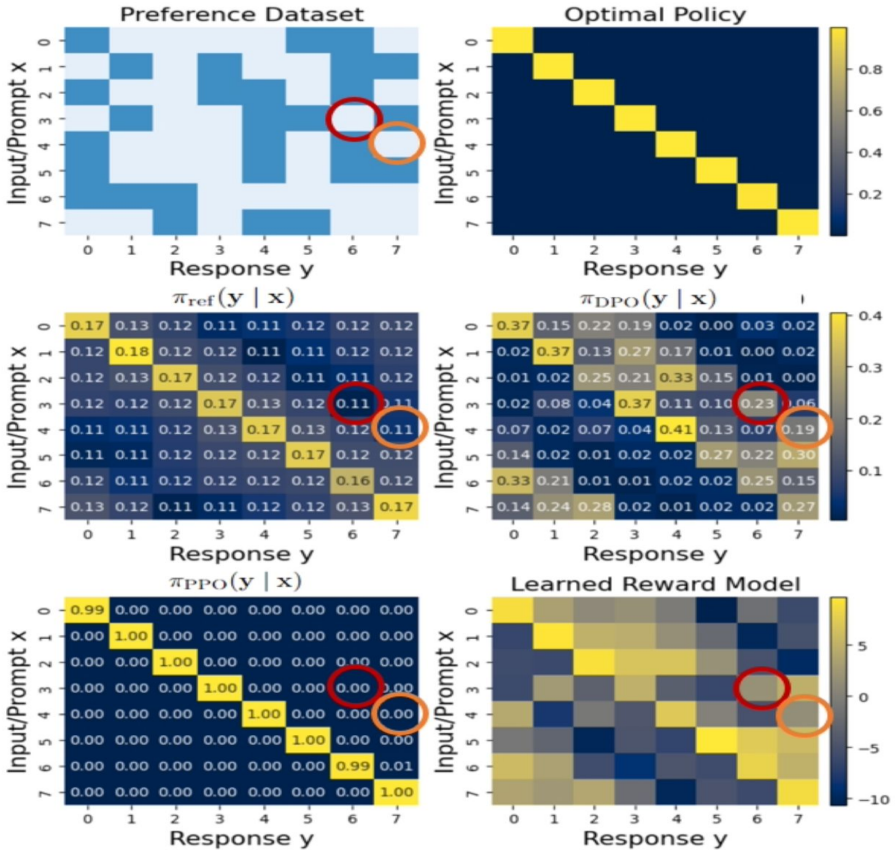
    → **Training** 단계에서 **생성된 샘플들에 KL-Divergence**를 통한 추가적인 **regularization**을 제공할 수 있음

4. DPO의 한계 - Synthetic Scenario을 통한 실험적 검증



- y축: Input/Prompt
- x축: Response Y
- policy & reward model MLP로 모델링
  - input: one-hot vector
  - output: 전체 답변에 대한 categorical dist'N
- 최적 답변들은 대각선 diagonal을 이루게끔 설정

4. DPO의 한계 - Synthetic Scenario을 통한 실험적 검증



- DPO & 학습된 Reward Model은 Preference Dataset의 Out-of-Dist'N 답변들에 대하여 높은 값을 매김. "Reward Misspecification"
  - DPO의 경우, reference model보다 높은 값을 매길 가능성 존재
- 반면 PPO는 낮은 값을 매겼음 & optimal policy를 학습함.
  - 앞서 Reward Model이 misspecification 문제가 있었음에도, PPO가 reference model과의 KL-Regularization을 통해 완화 시킬 수 있음

⇒ DPO는 Out-of-Dist'N 답변들을 편애하는 편향된 policy를 생성하는 경향이 있음
→ 이는 예측 불가한 행동을 초래할 수 있음

4. DPO의 한계 - 실제 Preference Dataset에 실험

DPO 성능에 영향을 끼칠 만한 두가지 factor 분석

- **Base Model & Preference Data**

실험 설계

- 데이터셋: **SafeRLHF……Preference 쌍 ($x, y_1, y_2, l_h, l_s, b_1, b_2$)**
  - **l_h, l_s : helpfulness & safety을 기준으로 한 y_1, y_2에 대한 선호도**
  - **b_1, b_2 : y_1, y_2에 대한 binary safety 라벨 (positive or negative)**

  → **Objective:** 내용 생성에 있어서 **helpfulness**보다 **safety**를 우선시하도록 **LLM** 훈련
  - 두 답변 모두 **safe**일때는 **helpfulness**에 따른 선호도, 그 외는 **safety**에 따른 선호도
    - **i.e. pref== If) b_1, b_2 모두 positive →l_h else) l_s**
- **base model**은 **Alpaca**에 **train w/SFT ⇒ 'SFT(Alpaca)'**

4. DPO의 한계 - 실제 Preference Dataset에 실험

Base Model의 영향

- SFT(Alpaca): DPO 성능 poor

  → 가설 : base model의 training data(Alpaca)와 preference data(SafeRLHF) 간의 분포 이동 (distribution shift)으로 인한 distribution mismatch & noises

- SFT(Safe): SFT(Alpaca)를 SafeRLHF 데이터셋의 safe 답변들만으로 fine-tune

  ⇒ 추가적인 SFT 이후의 DPO를 통해 분포 불일치 완화로 Safety Rate 16.4% 향상

Preference Data에 대한 Sensitivity

- dataset 정제를 통해서도 성능 향상을 보임

	ΔHelp. ↑	Harm. ↓	S.R. ↑
SFT (Alpaca)	-2.62	1.50	41.6%
PPO	1.69	-12.08	99.5%
+ SFT (Safe)	4.47	-12.33	99.6%
DPO	-4.19	-0.97	55.4%
+ SFT (Safe)	-1.62	-3.50	71.8%
+ filter dual-unsafe	2.46	-4.88	80.8%
+ filter dual-safe	-2.86	-6.82	95.8%
DPO Iter.1	-3.22	-5.23	86.7%
DPO Iter.2	-3.27	-8.83	99.7%
DPO Iter.3	-3.26	-10.21	99.9%
DPO Iter.4	-2.96	-11.07	99.9%

4. DPO의 한계 - 실제 Preference Dataset에 실험

Preference Data Dist'N의 영향

- 추가적인 **SFT** 이외의 **dist'N shift** 완화 방법 조사
- 추가적인 데이터 수집 방안
  - **SFT(Safe)**로 새로운 답변 생성 **&** 학습된 **reward model**이 **preference labeling**
  - **Iterative** 반복적으로 위의 과정 진행; 가장 마지막의 **DPO model**을 **reference model**로 설정

⇒ **DPO-Iter**

- **PPO**와 견줄만한 성능 보임

	ΔHelp. ↑	Harm. ↓	S.R. ↑
SFT (Alpaca)	-2.62	1.50	41.6%
PPO	1.69	-12.08	99.5%
+ SFT (Safe)	4.47	-12.33	99.6%
DPO	-4.19	-0.97	55.4%
+ SFT (Safe)	-1.62	-3.50	71.8%
+ filter dual-unsafe	2.46	-4.88	80.8%
+ filter dual-safe	-2.86	-6.82	95.8%
DPO Iter.1	-3.22	-5.23	86.7%
DPO Iter.2	-3.27	-8.83	99.7%
DPO Iter.3	-3.26	-10.21	99.9%
DPO Iter.4	-2.96	-11.07	99.9%

4. DPO의 한계 - 실제 Preference Dataset에 실험

DPO 성능 향상

- **model과 preference dataset사이의 분포 차이 혹은 이동을 완화**

  → i) 추가적인 SFT

  ii) noisy한 preference 데이터 정제

  iii) 'DPO-iter' - 새로운 response 생성& 학습된 reward model을 통한 labeling

5. PPO의 중요 요소

3개의 key technique

1. **Advantage Normalization**
    a. **advantage estimates를 정규화**
        i. **advantage function A(a,s): s(state)에서 a(action)을 취했을때 a들의 평균보다 결과값이 얼마나 잘하거나 못하는지 수치화**
2. **Large batch-size training**
3. **Exponential Moving Average를 통한 reference model 업데이트**
    a. '지수 평활법'
        i. 시계열 데이터를 매끈하게 만드는 경험 법칙

5. PPO의 중요 요소

Implementation 세부사항

- **PPO → DeepSpeed-Chat 기반**

Experiment Setup

- **Ablation 연구**
  - **Dialogue 대화 task → HH-RLHF task**
    - **Base 모델: Llama2-7B**
  - **코드 생성 task → APPS, CodeContest**
    - **Base 모델: CodeLlama-34B**

5. PPO의 중요 요소

Task	HH-RLHF	APPS			CodeContest		
Metric	OpenAssaint Reward	Intro. pass@5	Inter. pass@5	Comp. pass@5	pass@10	pass@100	pass@1k
SFT	0.532	38.6%	10.1%	3.9%	0.9%	4.3%	12.0%
baseline PPO	0.706	18.0%	2.4%	1.1%	4.3%	6.0%	7.7%
+ Adv.Norm.	0.716	38.1%	11.4%	4.6%	6.8%	9.4%	15.4%
+ Large.Batch.	0.716	42.3%	14.6%	7.5%	5.1%	12.8%	19.6%
+ Ref.EMA	**0.718**	44.4%	18.0%	9.1%	6.8%	13.7%	21.4%

- **Baseline PPO는 HH-RLHF & CodeContest에서는 성능 향상, APPS에서는 성능 저하를 보임**
- **Adv.Norm: PPO training 안정화 → PPO 성능 향상**
- **Large.Batch: 가장 비약적인 성능 향상을 불러옴, 특히 코드 생성 task들**
- **Ref.EMA (Reference 모델에 EMA): 추가적인 성능 향상, reference model 또한 PPO의 main LLM에 따른 update**

6. Benchmark 결과 - HH-RLHF

	OpenAssistant Reward	Tested V.S. Chosen			Tested V.S. SFT		
		Tested Win ↑	Tie	Chosen Win ↓	Tested Win ↑	Tie	SFT Win ↓
RRHF	0.523	28	33	39	29	37	34
PRO	0.529	37	26	37	34	33	33
DPO	0.611	55	21	24	53	31	16
DPO-Iter	0.678	55	18	27	54	33	13
PPO	**0.718**	57	21	22	58	29	13

Table 4. Results on the HH-RLHF test set. The evaluation metrics include the OpenAssistant rewards and the win rate of models against the chosen responses and SFT model outputs. The OpenAssistant reward model is not used during the training process. Note that DPO is trained on the preference data in the dataset, while Iter. DPO is trained on self-generated responses, using a reward model for labeling.

***LLama2-7B, OpenAssistant Reward Model**

⇒ PPO > DPO-Iter > DPO

6. Benchmark 결과 - SafeRLHF

LLM	Method	ΔHelp. ↑	Harm. ↓	S.R. ↑
	Beaver	-	-6.59	89.6%
Llama 1 7B	SFT	-2.26	0.78	46.5%
	DPO	-2.70	-6.38	93.1 %
	DPO-Iter	-2.79	**-11.86**	**100.0%**
	PPO	**+0.66**	-10.22	98.6%
Llama 2 7B	SFT	-2.12	0.00	52.1%
	DPO	-2.86	-6.82	95.8%
	DPO-Iter	-2.96	-11.07	**99.9%**
	PPO	**+1.69**	**-12.08**	99.5%

Table 6. Results on SafeRLHF. "Beaver" is the officially released model. "Δ Help." denotes helpfulness relative to Beaver. "S.R." denotes safety rate. The reported results are based on the official evaluation model.

⇒ **Alignment** 이후에, **DPO**와 **PPO** 모두 **harm**이 적은 답변 생성 가능

⇒ **PPO**의 답변이 더 **Helpful**

6. Benchmark 결과 - Apps

⇒ **CodeLlama-34B를 base model로 사용했을때 SOTA**

⇒ 모든 모델 **size**에서 **DPO-Iter**가 **SFT**보다 낮은 성능 보임

⇒ **PPO**는 모델 **size** 커질수록 성능 향상

Model	Method	Intro.	Inter.	Comp.
GPT-Neo 2.7B	SFT	5.6%	0.8%	0.0%
Codex 12B	SFT	9.7%	0.5%	0.1%
CodeT5	CodeRL	16.4%	4.9%	2.8%
AlphaCode 1B	5@1k	14.4%	5.6%	4.6%
Code Llama 7B	Few shot	10.8%	2.0%	0.8%
	SFT	**30.0%**	**7.8%**	**2.8%**
	DPO-Iter	20.9%	3.4%	1.3%
	PPO	29.4%	7.6%	2.4%
Code Llama 13B	Few shot	23.7%	5.6%	2.1%
	SFT	33.7%	8.7%	3.6%
	DPO-Iter	33.0%	8.0%	2.8%
	PPO	**36.4%**	**11.47%**	**4.6%**
Code Llama 34B	Few shot	32.8%	8.8%	2.9%
	SFT	38.6%	10.1%	3.9%
	DPO-Iter	34.2%	9.3%	3.7%
	PPO	**44.4%**	**18.0%**	**9.1%**

Table 7. Results on Apps test set. All the numbers are pass@5 except for AlphaCode. Where "5@1k" means this model samples 1000 times for each problem and 5 sampled codes that pass the public test cases (in the problem description) are selected to be evaluated on hidden test cases.

6. Benchmark 결과 - CodeContest

Model	Method	Valid. Set 10@1k	Test Set 10@1k
AlphaCode 9B	-	16.9%	14.3%
AlphaCode 41B	-	16.9%	15.6%
	+ clustering	21.0%	16.4%
Code Llama 34B	SFT	10.3%	15.2%
	DPO	0.0%	0.0%
	DPO-Iter	3.5%	3.2%
	PPO	19.7%	**22.4%**

Table 8. Pass rate on CodeContests dataset. "10@1k" means that 1000 samples will be evaluated on public tests in the problem description, and only 10 of them will be submitted for hidden tests. We **only used Python for solving problems**, while AlphaCode used both Python and C++.

⇒ **PPO**는 **SFT** 모델의 성능을 상당히 향상시킴

⇒ **CodeLlama-34B**는 **PPO**의 도움으로 **SOTA** 기록

⇒ **DPO**는 옳은 코드를 하나도 생성하지 못함

⇒ **DPO-Iter** 또한 **SFT**모델보다 못함

7. Conclusion

- DPO의 한계 분석
  - DPO는 base model output과 preference data 간의 Distribution shift에 민감
  - Iterative DPO 제시
- PPO Training의 Critical Factor 분석
  - Advantage normalization, large batch size, Exponential Moving Average를 통한 Reference 모델 업데이트
- 다양한 task에서의 PPO의 꾸준히 높은 성능
  - CodeCompetition 테스크에선 SOTA

Toward Informal Language Processing: Knowledge of Slang in Large Language Models

Zhewei Sun[1], **Qian Hu**[2], **Rahul Gupta**[2], **Richard Zemel**[2,3], **Yang Xu**[1]

[1]University of Toronto, [2]Amazon Alexa AI, [3]Columbia University
{zheweisun, yangxu}@cs.toronto.edu, {huqia, gupra}@amazon.com
zemel@cs.columbia.edu

2024 NAACL

심규호

Natural Language Processing & Artificial Intelligence

고려대학교 KOREA UNIVERSITY

0. Abstract

Slang 속어

- 일상의 대화 **& online** 소셜미디어에서 흔히 사용됨
- 정밀하게 짜여진 **public benchmark**의 부재

⇒ 영화 자막 **Movie Subtitles**들을 통해 **Evaluation & Fine-tuning**을 위한 **Dataset** 구축

1) **Slang Detection**
2) 문장에서의 **slang**에 대한 지역적 **&** 역사적 정체성 파악

1. Introduction

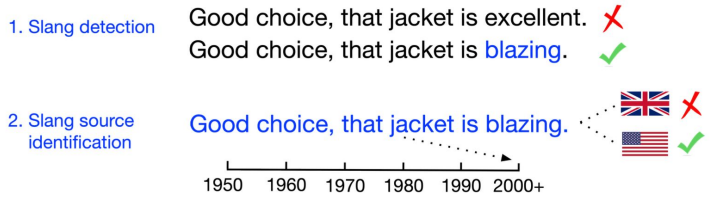


Figure 1: Overview of tasks used to probe knowledge of slang in LLMs.

- **LLM의 Slang**에 대한 지식 부족
- **Slang**의 중요성
  - 사회적 정체성을 나타냄 **e.g.** 나이, 성별,
    - 'blazing': US; 'Something Excellent', UK; "Anger"

⇒ 이러한 **Slang**의 중요성에도 불구하고, **slang**과 관련된 **LLM** 연구는 적음

- 특히 높은 퀄리티의 **public dataset** 부족

⇒ **'OpenSubtitles' corpus** 기반으로 **publicly accessible dataset** 구축

i) LLM의 Slang 검출 능력

ii) task-classification - 문장에서의 **slang**에 대한 지역적 **&** 역사적 정체성 파악

iii) Semantic Knowledge of slang in LLMs to understand the difference in the representation of slang versus conventional language use

전통적인 언어 사용과 **Slang**의 차이를 이해할 수 있는 **Slang**에 대한 의미적 지식

1. Introduction - Contributions

1. 영화 자막에 사용된 **human-annotated** 영어 **Slang**을 포함한 **publicly accessible dataset**
    a. **Novelty -** 기존에 없던 **public** 접근 가능한 **Slang benchmark** 제공
2. **LLM**의 **Slang** 관련 지식에 대한 견고한 평가

2. Related Work

Deep Learning for Slang

- **Detection, generation, interpretation, predicting word formations of slang**
  → **Slang의 데이터 부족으로 연구에 어려움**

Probing Knowledges in LLMs

- **LLMs on their knowledge of non-standard language**
  - **e.g. metaphors, linguistic anomalies**
- 두 가지 **probing framework**

  i) Behavioral Probing: 유사 input에 대한 언어 모델의 다른 행동 분석

  ii) Edge Probing: 텍스트의 적절한 범위에 있는 모든 토큰의 표현을 집계하여 레이블을 예측 방식

  ⇒ Behavioral Probing for LLM's confidence in predicting slang usage
  - 같은 문맥에 사용되는 **Slang과 literal token의 LM 확률 비교**

  ⇒ Edge Probing for Slang detection & Slang Source Identification
  - **LLM의 속어 사용과 인구통계학적 정체성에 대한 지식**

3. Data

Open-Sub Slang Dataset

Dataset	Slang detection and probing			Slang source identification		Publically accessible
	Slang-containing sentences	Non-slang sentences	Word-level paraphases	Community of emergence	Time of emergence	
Urban Dictionary	✓	✗	✗	✗	✗	✓
The Online Slang Dictionary (OSD)	✓	✗	✓	✗	✗	✗
Green's Dictionary of Slang (GDoS)	✓	✗	✗	✓	✓	✗
Reddit Glossaries (Lucy and Bamman, 2021)	✗	✗	✗	✓	✗	✓
Indonesian Colloquialism (Wibowo et al., 2021)	✗	✗	✓	✗	✗	✓
OpenSubtitles-Slang (OpenSub-Slang)	✓	✓	✓	✓	✓	✓

Table 1: Summary of datasets for slang in NLP and the availability of important features for a comprehensive benchmark. We contribute a new resource (OpenSub-Slang) that captures all desirable features.

4. Experiments - Models

Models

- BERT- like models
  - BERT(bert-large-cased), RoBERTa(roberta-large), XLNet(xlnet-large-cased)
- GPT models
  - GPT-3(text-davinci-002),GPT-3.5(gpt-3.5-turbo-0613),
    GPT-4(gpt-4-1106-preview)

4. Experiments - Slang Detection

Task

- Edge Probing: 두개의 **slang detection** 테스크
  - **3개의 Bert-like masked-language models**
  - **zero-shot & fine-tune 된 GPT 모델들**
    - **zero-shot; GPT의 내재된 지식 탐구하기 위함**
- 문장-단위: **binary classification**
  - **slang 포함이면 +, 포함하지 않으면**
- 단어-단위: **sequence tagging task**
  - **slang인 단어에 *slang* label, 그외는 null**

(S1) Good choice, that jacket is **blazing**.

(S2) Good choice, that jacket is excellent.

4. Experiments - Slang Detection

Results

- OpenSubtitles-Slang dataset의 문장들에 slang detection 결과
- BERT, RoBERTa >> XL-Net
- fine-tuned GPT-3.5 >> BERT-like
- fine-tuned BERT, RoBERTa >= zero-shot GPT
    - 훨씬 적은 파라미터인 점
- Word-Level: GPT-finetuned >> GPT zero-shot models

⇒ GPT model들이 slang을 detect하는 관련 지식들이 더 많지만, GPT또한 더 나은 성능을 위해서 fine-tuning 과정이 필요

(a) Sentence-Level detection

Model	P	R	F1
BERT	80.1	83.3	81.6
RoBERTa	81.3	87.5	84.2
XL-Net	67.5	64.3	64.6
GPT-3 zero-shot	**90.0**	74.4	81.4
GPT-3.5 zero-shot	87.5	80.8	84.0
GPT-4 zero-shot	88.2	80.9	84.4
GPT-3.5 finetuned	84.3	**96.8**	**90.1**

(b) Word-Level detection

Model	P	R	F1
BERT	75.5	62.5	68.3
RoBERTa	74.9	68.2	71.4
XL-Net	62.4	43.3	51.0
GPT-3 zero-shot	49.2	59.9	54.0
GPT-3.5 zero-shot	57.6	73.2	64.5
GPT-4 zero-shot	60.4	68.2	64.1
GPT-3.5 finetuned	**74.5**	**81.3**	**77.8**

Table 2: Slang detection results of LLMs shown in precision (P), recall (R), and F1 Scores (F1).

4. Experiments - Slang Detection

Results

- 영국 **UK**의 **slang**이 미국보다 더 자주 모델에 의해 검출됨
- 눈에 띄는 성능 차이

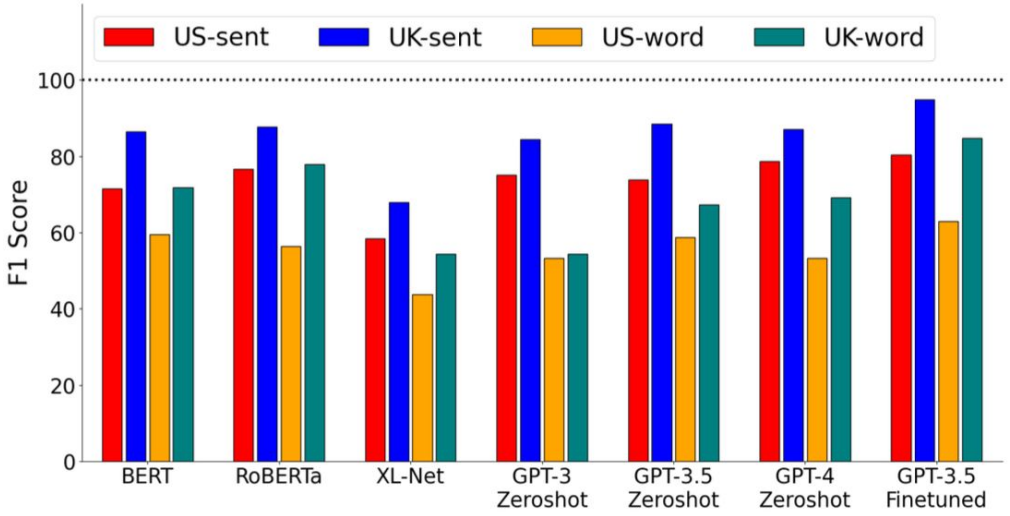


Figure 2: Slang detection performance by region.

4. Experiments - Slang Source Identification

Task

LLM의 slang의 인구통계학적 정체성 관련 지식 탐구 → text classification task 수행

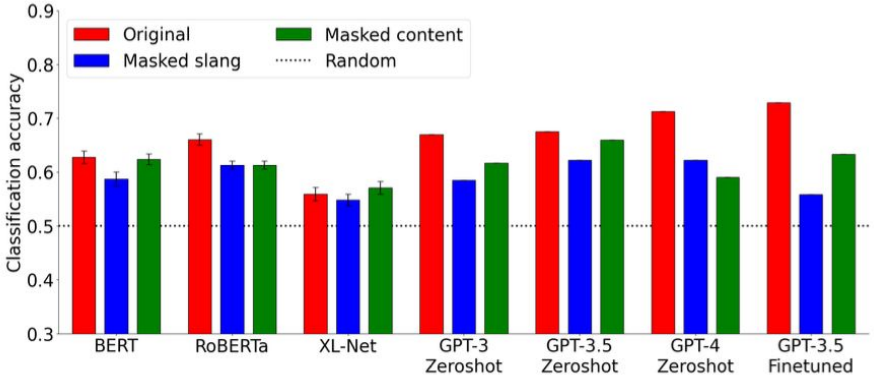
- **Slang을 포함한 문장을 주고 모델에게 해당 Slang의 원천을 classify하게끔 하는 task**
- 가설
  - 모델에게 있어서 인구통계학적 원천을 알아내는데 속어가 **salient feature**라면, **S2**와 같이 속어가 **mask**된 경우 현저히 떨어진 성능을 보일것
  - 반면, **random** 단어 **mask**한 것은 그정도의 성능 저하는 보이지 않을것

(S1) Good choice, that jacket is **blazing**.

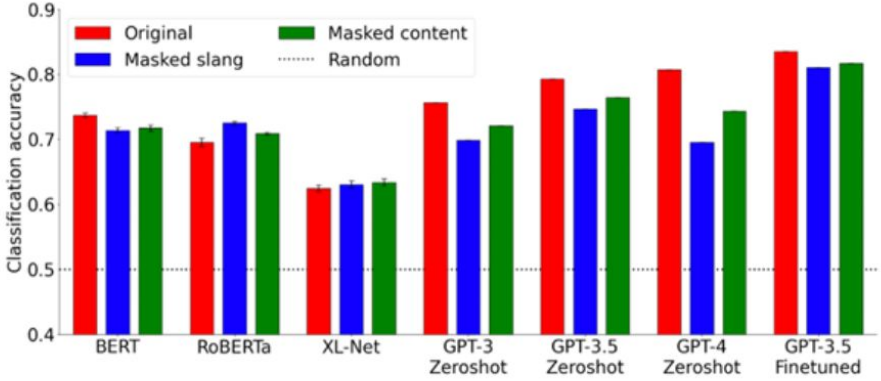
(S2) Good choice, that jacket is [MASK].

(S3) Good [MASK], that jacket is blazing.

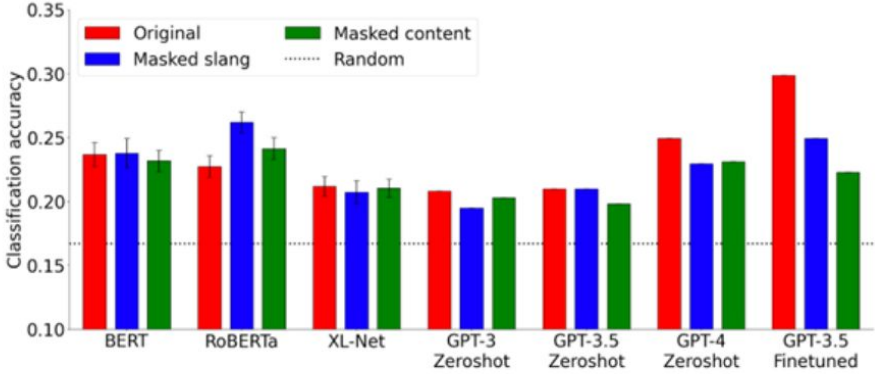
(a) OpenSubtitles-Slang - Region



(b) Green's Dictionary of Slang - Region



(c) Green's Dictionary of Slang - Age



Salient feature

⇒ GPT는 slang이 salient feature (O)

- 특히 **GPT-3**에서는 명백히 관측됨

⇒ 반면 BERT는 slang이 salient feature (X)

- **Slang**이 **mask**될 때나 **Random word**이 **mask**될 때나 비슷한 성능 저하 → 단어의 삭제 자체가 성능 저하를 일으킴
- **Slang**이 **mask**될 때 오히려 성능 향상되는 경우도 존재 → **BERT** 모델들은 오히려 **slang**이 삭제되고 나서야 문장의 뜻에 더 확신을 가짐

4. Experiments - Model Interpretation

모델이 **Slang**에 대한 구조적 이해를 하는지

- **slang**과 그에 상응하는 **paraphrase token**의 확률 비교
- 문장 **embedding** 분석

$$r_{mean} = \frac{\sum_i \mathcal{S}_i}{\sum_i \mathcal{L}_i}$$

$$r_{median} = \text{median}_i \frac{\mathcal{S}_i}{\mathcal{L}_i}$$

Task

- **model**의 **slang** 사용에 대한 **predictive confidence**

Metrics

slang과 그의 **literal counterpart**의 **predictive confidence** 비교를 위한 두 개의 **metric**

- let) S_i …i번째 문장의 **slang**에 지정된 언어모델 확률
  L_i…i번째 문장의 **literal counterpart**에 지정된 언어모델 확률

(S1) Good choice, that jacket is **blazing**.

(S2) Good choice, that jacket is **excellent**.

4. Experiments - Model Interpretation

⇒ 오른쪽 표는 Slang과 Literal Token의 likelihood ratio

⇒ BERT, RoBERTa, XL-Net, GPT-3 모두 median ratio > mean ratio

→ 많은 slang들에 대해 confident하지만 특정 subset의 slang들에 대해선 지식이 부족

$$r_{mean} = \frac{\sum_i \mathcal{S}_i}{\sum_i \mathcal{L}_i} \quad r_{median} = \text{median}_i \frac{\mathcal{S}_i}{\mathcal{L}_i}$$

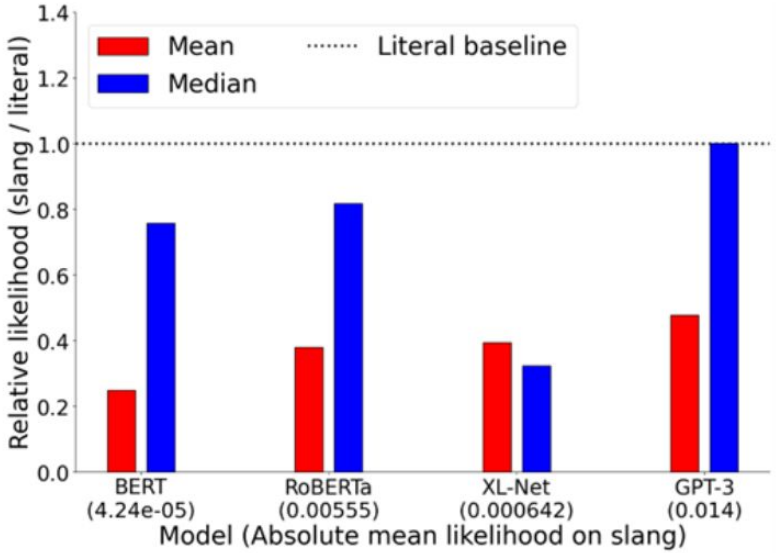


Figure 5: Likelihood ratios between samples of corresponding slang and literal tokens.

4. Experiments - Model Interpretation

⇒ 왼쪽 표는 US & UK별 Median Ratio

⇒ BERT, RoBERTa, XL-Net, GPT-3 모두

US median ratio > UK median ratio

→ US slang을 생성하는게 UK Slang을 생성하는것보다 훨씬 confident

$$r_{mean} = \frac{\sum_i \mathcal{S}_i}{\sum_i \mathcal{L}_i} \quad r_{median} = \text{median}_i \frac{\mathcal{S}_i}{\mathcal{L}_i}$$

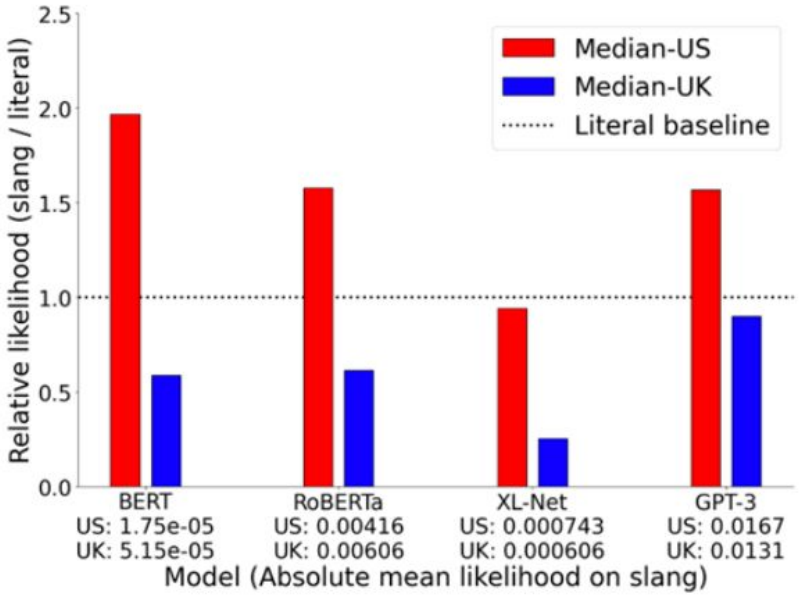


Figure 6: Median ratios across sentences from different regions.

5. Conclusion

⇒ 거대 **GPT** 모델들이 **BERT**기반의 모델들 보다 **Slang** 관련 지식들을 더 많이 갖고 있음

i) 문장에서의 **slang** 검출 능력이 더 뛰어남

ii) slang의 시대/지역 요소를 더 정확하게 파악함

iii) Better predicting slang usages relative to their literal counterparts

*이러한 **GPT**의 우위에도 불구하고, 해당모델이 **Slang**을 특별한 언어적 형태로 **encoding**했다는 증거는 찾지 못함

- 영국의 **Slang &** 더 최근에 나온 **slang**들에 모델들이 더 떨어진 성능을 보임 → **data** 부족의 원인으로 추정됨

⇒ **GPT**는 **BERT**모델들과 비슷한 **bias & processing of slang**

⇒ **GPT model**들이 문맥별 속어 사용의 인구통계학적 정체성들을 더 잘 파악

5. Limitations

- **GPT-Models**에 직접적인 접근 권한 통제
  - **GPT**의 **internal layer**들 접근 불가 → 비교 방해
    - **e.g. GPT-3**만 확률값 분석 가능, 그 이상의 모델은 불가능
- **GPT; Autoregressive**한 성질은 **BERT;MLM**한 특징과 부딪힘 → 비교 방해
- 해당 연구는 언어를 영어로만 했다는 점과 인구통계학적인 군집단도 한정됨

감사합니다