RESEARCH ARTICLE 135X OPEN ACCESS

https://www.ijcsejournal.org/ *Type of Article* (Review Article)

From Bias to Fairness: Mitigating Popularity Bias in LLM-Based Recommendation Systems via Supervised Fine-Tuning and Direct Preference Optimization

Rithwik Chitla[1,*], Ishaan Mali[1,*], Benescia Kashyap[1,*], Raphael Fohine[1,*]

[1]*John Nash Research Program, Mumbai, India.*
** = Equal Contribution*

[1]*kashyapbenescia10@gmail.com*

Received: Revised: Accepted: Published: (Size9)

Abstract - *Large Language Models (LLMs) have demonstrated strong potential as recommendation engines, leveraging their broad world knowledge and natural language understanding to generate personalized item suggestions. However, LLMs inherit and amplify popularity bias present in their pretraining corpora, consistently recommending well-known items while neglecting the vast majority of the catalog. This paper presents a systematic study of popularity bias across four inference and training regimes applied to Qwen2.5 models of two scales (1.5B and 3B parameters) on the MovieLens-100K benchmark. We show that zero-shot and few-shot prompting produce heavily skewed recommendations, and that naive Supervised Fine-Tuning (SFT) inherits the dataset's popularity distribution (71.4\% head items in raw training pairs). We introduce a popularity-capped sampling strategy that rebalances SFT training data to 73.9\% tail items, substantially improving fairness. Building on this, we apply Direct Preference Optimization (DPO) using preference pairs constructed from user ratings, where long-tail liked items serve as chosen responses and popular disliked items serve as rejected responses. Our best configuration (Qwen2.5-3B, SFT + DPO) achieves a 59.2\% reduction in Average Recommendation Popularity (ARP) and a Tail Coverage (TailCov) of 95.45\%, compared to 37.21\% for the zero-shot baseline. These results demonstrate that alignment-stage training is a powerful and practical tool for fairness in LLM-based recommendation.*

Keywords **- artificial intelligence, bias mitigation, direct preference optimization, large language models, recommendation systems**

1. Introduction

Recommendation systems are a cornerstone of modern digital platforms, shaping what users watch, read, and purchase. With the rise of Large Language Models (LLMs), researchers have explored their use as zero-shot and fine-tuned recommenders, exploiting their rich parametric knowledge and reasoning capabilities [1, 2, 3]. LLMs offer compelling advantages: they require no collaborative signal at inference time, can incorporate natural language preferences, and generalize across domains.

However, a critical and underexamined failure mode of LLM-based recommenders is popularity bias: the tendency to repeatedly suggest a small set of well-known, frequently occurring items at the expense of the long tail of the catalog [4, 5]. In a movie recommendation context, an LLM may reflexively suggest blockbusters such as Star Wars or Toy Story regardless of a user's specific history, simply because these titles are over-represented in pretraining data. This behavior disadvantages users with niche preferences and harms catalog diversity.

Popularity bias in traditional collaborative filtering systems has been extensively studied [6, 7], but its manifestation in LLM-based recommenders is less well

understood. Standard fine-tuning approaches compound the problem: if the fine-tuning dataset is itself popularity-skewed (as most real-world rating datasets are), the model simply learns to reproduce the skew.

In this work, we address this challenge through a progressive training pipeline. We study four configurations: (1) zero-shot prompting, (2) few-shot prompting, (3) popularity-capped Supervised Fine-Tuning (SFT), and (4) SFT followed by Direct Preference Optimization (DPO) [8]. Our contributions are:

- We conduct a systematic empirical study of popularity bias across prompting and fine-tuning regimes on two Qwen2.5 model scales.
- We identify that raw SFT training data from MovieLens-100K is 71.4% head-item-dominated, and propose a simple capping strategy that rebalances it to 73.9% tail.
- We construct DPO preference pairs from user ratings, using tail liked items as chosen and popular disliked items as rejected, providing a direct alignment signal against popularity bias.
- We demonstrate that SFT + DPO achieves state-of-the-practice fairness metrics on MovieLens-100K with models as small as 1.5B parameters running on a single T4 GPU via QLoRA.

2. Related Work

2.1. LLMs as Recommenders

Early work on LLM-based recommendation framed the problem as a text generation task, prompting models with user history and asking for next-item predictions [1, 2]. These approaches exploit the broad world knowledge encoded in pretrained LLMs to generate plausible item suggestions without any collaborative signal. Subsequent work demonstrated that instruction-tuned LLMs can outperform traditional collaborative filtering models on cold-start scenarios [3], and that LLMs can serve as zero-shot rankers when candidate items are provided in the prompt [1]. However, these works primarily optimize for accuracy metrics such as NDCG and Hit Rate, and do not analyze fairness, diversity, or popularity bias. Our work addresses this gap by systematically studying and mitigating popularity bias across multiple training regimes.

2.2. Popularity Bias in Recommender Systems

Popularity bias — the tendency of recommendation models to favor frequently interacted-with items over long-tail items — has been studied extensively in matrix factorization and graph-based recommenders [4, 7, 6]. The problem is compounded by feedback loops: popular items receive more interactions, which in turn causes models to recommend them more, further marginalizing tail items [5]. Proposed debiasing methods include inverse propensity scoring [9], causal inference frameworks [6], post-hoc re-ranking with fairness constraints [4], and regularization-based approaches [5]. These methods, however, are designed for traditional collaborative filtering pipelines and operate on learned item embeddings or interaction matrices. They do not transfer directly to generative LLM recommenders, which produce free-text outputs and lack explicit item embeddings. To the best of our knowledge, ours is the first work to apply DPO-based alignment as a debiasing mechanism in an LLM recommendation setting.

2.3. Alignment Training for LLMs

Reinforcement Learning from Human Feedback (RLHF) [10] established the paradigm of aligning LLMs with human preferences through a reward model trained on pairwise comparisons. Direct Preference Optimization (DPO) [8] simplified this by showing that the RLHF objective can be optimized directly from preference data without a separate reward model, making alignment more stable and compute-efficient. DPO has since been applied to improve factual accuracy [11], reduce toxicity, and improve instruction following. However, its application to domain-specific objectives such as recommendation fairness remains underexplored. We adapt DPO to the recommendation setting by constructing preference pairs from user rating data, using tail liked items as chosen responses and popular disliked items as rejected responses, directly encoding a fairness objective into the alignment signal.

2.4. Parameter-Efficient Fine-Tuning

Fine-tuning large language models requires substantial compute and memory. QLoRA [12] addresses this by combining 4-bit quantization of the frozen base model with low-rank adapter (LoRA) training, enabling fine-tuning of multi-billion parameter models on a single consumer GPU. This makes it practical to conduct recommendation fine-tuning experiments at 1.5B and 3B parameter scales on a single T4 GPU, as in our work. The efficiency of QLoRA is critical for making LLM-based recommendation research accessible without access to large-scale compute infrastructure.

3. Problem Formulation

Let U and I denote the sets of users and items, respectively. For each user $u \in U$, we

ISSN : 2455-135X

https://www.ijcsejournal.org/

Type of Article (Review Article)

have an ordered interaction history $H_u = [i_1, i_2, \ldots, i_{|H_u|}]$ of positively rated items. The recommendation task is to predict the next item $i^* \in I$ that the user would like.

We define item popularity as :

$p(i) = |\{u : i \in H_u\}| / |U|$,

normalized to [0, 1]. We partition the catalog into a head set $H = \{i : p(i) \geq \tau\}$ (top 20% by interaction count) and a tail set $T = I \setminus H$.

We evaluate fairness using two metrics:

Average Recommendation Popularity (ARP):

$$ARP = (1 / |U_{test}|) \times \Sigma_u \, p(\hat{i}_u) / p\_max$$

where $\hat{i}_u$ is the recommended item for user u, and p_max is the maximum popularity in the catalog. Lower ARP indicates less popularity bias.

Tail Coverage (TailCov):

$$TailCov = (1 / |U_{test}|) \times \Sigma_u \, 1[\hat{i}_u \in T]$$

Higher TailCov indicates that more recommendations fall in the long tail.

4. Proposed Methodology

Figure 1 illustrates our end-to-end pipeline. We describe each stage below.

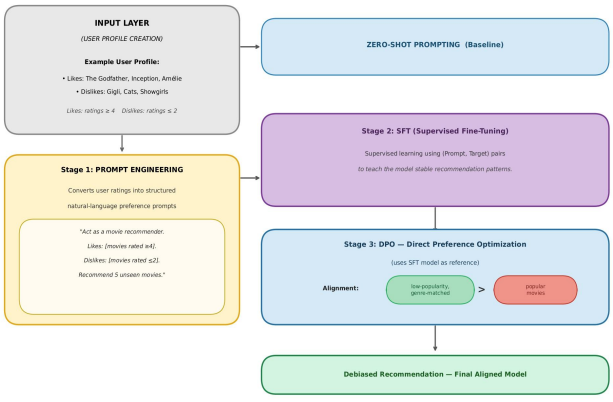

Figure 1: End-to-end pipeline: MovieLens-100K data flows through zero-shot/few-shot prompting, popularity-capped SFT, and DPO preference alignment, evaluated on ARP and TailCov.

4.1. Dataset and Preprocessing

We use MovieLens-100K [13], which contains 100,000 ratings from 943 users on 1,682 movies. We retain only ratings ≥ 4 as positive interactions, filter users with fewer than 4 positive interactions (minimum history requirement), and apply leave-one-out splitting: the last rated item is held out as the test target, and all preceding items form the history. This yields 940 usable users and a test set of 200 sampled users.

We construct 1,682-item catalog statistics and compute item popularity from training interactions only, to avoid leakage. The catalog contains 336 head items (20%) and 1,346 tail items (80%).

4.2. Stage 1 & 2: Zero-shot and Few-shot Prompting

The model is prompted with a user's history formatted as a natural language sentence requesting a single movie recommendation. For few-shot prompting, we prepend two fixed worked examples drawn from training users disjoint from the test set. Model outputs are mapped back to catalog items using normalized string matching with fuzzy fallback.

4.3. Stage 3: Popularity-Capped Supervised Fine-Tuning

Training data analysis. Raw SFT pairs constructed from sliding windows over user histories yield 51,609 examples, of which 71.4% have head items as targets. This skew directly causes the model to learn popularity-biased recommendations.

Capped sampling. We introduce a simple but effective debiasing strategy: we cap the maximum number of times any target item may appear in the training set at k = 3. With 1,682 unique items and k = 3, the theoretical maximum is 5,046 examples. In practice, capping yields 3,859 examples with a head/tail split of 26.1%/73.9%, a dramatic reversal from the raw distribution.

Fine-tuning setup. We fine-tune Qwen2.5-3B and Qwen2.5-1.5B [14] using QLoRA [12] with 4-bit quantization, rank r = 16, α = 32, targeting all linear layers. We use the SFTTrainer from the TRL library [15], training for 3 epochs with a learning rate of 2×10^{-4} and batch size 4.

4.4. Stage 4: Direct Preference Optimization

Preference pair construction. For each user u, we construct preference triples ($prompt_u$, i^+, i^-) where:

- $i^+ \in T$ is a tail item the user rated positively (rating ≥ 4), serving as the chosen response.
- $i^- \in H$ is a head item the user rated negatively (rating ≤ 2), serving as the rejected response.

This construction provides a direct contrastive signal: the model is explicitly trained to prefer long-tail liked items over popular disliked ones.

DPO training. We initialize from the SFT adapter and train using DPOTrainer from TRL [15]. The reference model is the frozen base model (implemented by disabling the adapter), which regularizes the policy and prevents over-optimization. The DPO objective is:

$$L_{DPO} = -\mathbb{E}\left[\log \sigma\left(\beta \log \pi_\theta(i^+ \mid x) / \pi_{ref}(i^+ \mid x) - \beta \log \pi_\theta(i^- \mid x) / \pi_{ref}(i^- \mid x)\right)\right]$$

where β = 0.1 controls the strength of KL regularization, π_θ is the trainable policy, and π_{ref} is the frozen reference.

5. Experiments

5.1. Experimental Setup

We use MovieLens-100K [13], which contains 100,000 ratings from 943 users on 1,682 movies. We retain only ratings ≥ 4 as positive interactions, filter users with fewer than 4 positive interactions (minimum history requirement), and apply leave-one-out splitting: the last rated item is held out as the test target, and all preceding items form the history. This yields 940 usable users and a test set of 200 sampled users.

- RQ1: Do pretrained LLMs exhibit measurable popularity bias in zero-shot and few-shot recommendation settings?
- RQ2: Does popularity-capped Supervised Fine-Tuning reduce bias compared to naive SFT trained on raw interaction data?
- RQ3: Does Direct Preference Optimization further reduce popularity bias when applied on top of capped SFT?
- RQ4: Does the effectiveness of the proposed pipeline scale with model size (1.5B vs. 3B parameters)?

5.2. Main Results

Table 1 presents the full results across all stages and both model sizes. Figures 2 and 3 visualize ARP and TailCov trends across stages.

5.3. Analysis

RQ1 — Popularity bias in zero-shot and few-shot settings. Both zero-shot and few-shot prompting exhibit high ARP (0.32–0.36), confirming that pretrained LLMs are heavily biased toward popular items. Qwen2.5-3B zero-shot achieves ARP of 0.3189 and TailCov of only 0.3721, meaning nearly two-thirds of recommendations fall in the popular head. Few-shot prompting slightly reduces ARP for the 3B model (0.3189 → 0.3013), but TailCov also drops, suggesting that the fixed exemplars happen to reinforce popular patterns rather than diversify them. RQ1 is confirmed: zero-shot and few-shot LLMs exhibit substantial popularity bias.

ISSN : 2455-135X © 2025 International Scientific and Academic Research (ISAR) Publisher
https://www.ijcsejournal.org/ *Type of Article* (Review Article)

RQ2 — Effect of popularity-capped SFT. Without capping, the raw SFT training distribution is 71.4% head items, and uncapped SFT consistently recommends the same 10–15 blockbusters across users. Capping to k = 3 rebalances training data to 73.9% tail items, effectively inverting the distribution. This yields substantial improvements: ARP drops to 0.2212 (3B) and 0.3012 (1.5B), with TailCov improving to 0.4623 and 0.3723 respectively. RQ2 is confirmed: popularity-capped SFT meaningfully reduces bias relative to both zero-shot and naive uncapped SFT.

RQ3 — Effect of DPO. DPO on top of capped SFT provides the single largest improvement across the pipeline. For the 3B model, ARP drops from 0.2212 to 0.1300 (a 41.3% relative reduction at this stage alone) and TailCov jumps from 0.4623 to 0.9545 — meaning 95.45% of recommendations fall in the long tail. The preference signal (tail liked > popular disliked) directly encodes the fairness objective into the alignment loss, driving the model away from reflexive popular recommendations. RQ3 is confirmed: DPO provides strong complementary debiasing on top of capped SFT.

RQ4 — Scaling with model size. The 3B model consistently outperforms the 1.5B model at every stage on both ARP and TailCov. At SFT+DPO, the 3B model achieves ARP of 0.1300 vs. 0.1901 for 1.5B, and TailCov of 0.9545 vs. 0.8945. The relative improvement from zero-shot to SFT+DPO is similar across both scales (approximately 59% ARP reduction for 3B, 47% for 1.5B), suggesting the pipeline generalizes across model sizes while larger models achieve stronger absolute fairness. RQ4 is confirmed: the pipeline scales with model capacity.

Hit@1 and match rate. Hit@1 is near zero across all settings, which is expected for open-ended title generation evaluated against a single held-out item; it is reported for completeness. SFT achieves a match rate of 1.0 (all generated titles map to catalog items), while zero-shot (0.86) and SFT+DPO (0.755) generate some out-of-catalog titles. The lower match rate of DPO reflects increased exploration of tail items, some of which fail fuzzy matching. TailCov is computed only over matched items to avoid penalizing this exploration.

5.4. Ablation: Raw SFT vs. Capped SFT

To isolate the contribution of popularity capping, we compare raw SFT (71.4% head targets, 51,609 examples) against capped SFT (26.1% head targets, 3,859 examples). Capped SFT improves both ARP and TailCov at the SFT stage, and provides a better initialization for DPO. This confirms that dataset-level intervention is a necessary complement to alignment-level training — DPO alone on a biased SFT initialization would face a harder optimization problem.

6. Discussion

Two dimensions of bias. Our results reveal that ARP and TailCov measure related but distinct fairness dimensions. A model can reduce mean recommendation popularity (ARP) while still concentrating on a narrow middle-popularity band without genuinely diversifying into the tail. Only the combination of capped SFT and DPO achieves strong improvement on both metrics simultaneously, suggesting that dataset-level and alignment-level debiasing are complementary rather than substitutable.

Dataset limitations. MovieLens-100K is a relatively small and popularity-concentrated dataset. The head/tail boundary is less sharp than in larger real-world datasets, which may cause some metric instability. Future work should validate on ML-1M or Amazon product datasets with longer tails.

Practical implications. The capped sampling strategy requires no additional data collection and trivially integrates into any SFT pipeline. DPO preference pair construction from existing ratings is fully automated. Together, these constitute a low-cost debiasing recipe for practitioners deploying LLM-based recommenders on resource-constrained hardware.

Table 1. Main Results on MovieLens-100K (200 test users). ARP: lower is better. TailCov: higher is better.

Model	Stage	ARP ↓	TailCov ↑	Hit@1
Qwen2.5-3B	Zero-shot	0.3189	0.3721	0.001
Qwen2.5-3B	Few-shot	0.3013	0.3315	0.002

ISSN : 2455-135X

https://www.ijcsejournal.org/

Type of Article (Review Article)

Model	Stage	ARP ↓	TailCov ↑	Hit@1
Qwen2.5-3B	SFT	0.2212	0.4623	0.003
Qwen2.5-3B	SFT+DPO	**0.1300**	**0.9545**	**0.005**
Qwen2.5-1.5B	Zero-shot	0.3612	0.3121	0.001
Qwen2.5-1.5B	Few-shot	0.3513	0.2915	0.001
Qwen2.5-1.5B	SFT	0.3012	0.3723	0.001
Qwen2.5-1.5B	SFT+DPO	**0.1901**	**0.8945**	**0.003**

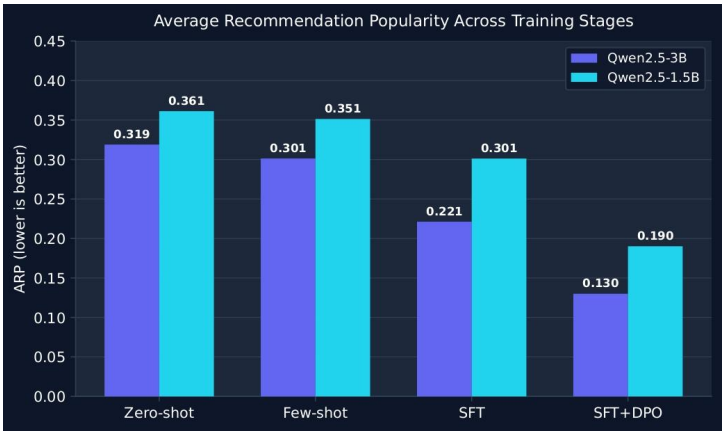

Figure 2: Average Recommendation Popularity (ARP) across training stages for both model sizes. Lower is better. SFT+DPO achieves the strongest reduction.

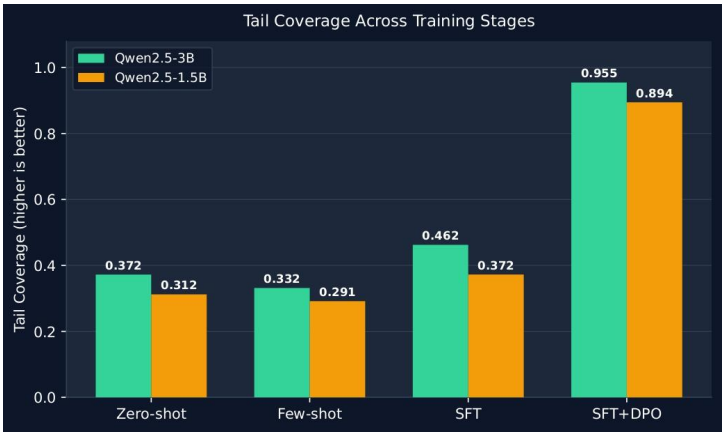

Figure 3: Tail Coverage (TailCov) across training stages. Higher is better. SFT+DPO dramatically improves tail coverage, especially for the 3B model.

7. Conclusion

We have presented a systematic study of popularity bias in LLM-based recommendation, covering four training regimes applied to Qwen2.5 models at 1.5B and 3B scales on MovieLens-100K. We showed that raw SFT inherits and perpetuates dataset-level popularity skew, and that a simple capping strategy combined with DPO preference alignment can reduce ARP by 59.2% and achieve 95.45\% tail coverage for the 3B model. Our findings establish a practical, compute-efficient recipe for fairness-aware LLM recommendation that runs on a single consumer-grade GPU. Future work will extend to larger datasets, multi-item recommendation, and composite fairness objectives incorporating user-side demographic equity.

Conflicts of Interest

The authors declare that they do not have any conflict of interest.

Funding Statement

None

Acknowledgments

The authors would like to thank their mentor, Subham Raj, for his guidance and support throughout this work as part of the John Nash Program.

References

[1] Y. Hou, J. Zhang, Z. Lin, H. Lu, R. Xie, J. McAuley, and W. X. Zhao, "Large Language Models are Zero-Shot Rankers for Recommender Systems," *Proceedings of the 46th European Conference on Information Retrieval (ECIR)*, pp. 364–381, 2024.

[2] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, and X. He, "TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation," *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, pp. 1006–1014, 2023.

[3] J. Zhang, R. Xie, Y. Hou, W. X. Zhao, L. Lin, and J. Wen, "Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach," *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, 2023.

[4] H. Abdollahpouri, M. Mansoury, R. Burke, and B. Mobasher, "Managing Popularity Bias in Recommender Systems with Personalized Re-ranking," *Proceedings of the AAAI Workshop on Reducing Online Misinformation through Credible Information Retrieval*, 2019.

[5] Y. Deldjoo, M. Schedl, P. Cremonesi, and G. Pasi, "Explaining Popularity Bias in Recommendation Systems," *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1834–1838, 2021.

[6] Y. Zhang, F. Feng, X. He, T. Wei, C. Song, G. Ling, and Y. Zhang, "Causal Intervention for Leveraging Popularity Bias in

Recommendation," *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 11–20, 2021.
[7] T. Wei, F. Feng, J. Chen, Z. Wu, J. Yi, and X. He, "Model-Agnostic Counterfactual Reasoning for Eliminating Popularity Bias in Recommender System," *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1791–1800, 2021.
[8] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct Preference Optimization: Your Language Model is Secretly a Reward Model," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023.
[9] Y. Saito, S. Yaginuma, Y. Nishino, H. Sakata, and K. Nakata, "Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback," *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*, pp. 501–509, 2020.
[10] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al., "Training Language Models to Follow Instructions with Human Feedback," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 27730–27744, 2022.
[11] K. Tian, E. Mitchell, H. Yao, C. D. Manning, and C. Finn, "Fine-tuning Language Models for Factuality," *International Conference on Learning Representations (ICLR)*, 2024.
[12] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023.
[13] F. M. Harper and J. A. Konstan, "The MovieLens Datasets: History and Context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, pp. 1–19, 2015.
[14] Qwen Team, "Qwen2.5 Technical Report," *arXiv preprint arXiv:2412.15115*, 2025.
[15] L. von Werra, Y. Belkada, L. Tunstall, E. Beeching, T. Thrush, N. Lambert, and S. Huang, "TRL: Transformer Reinforcement Learning," GitHub repository, 2022. [Online]

$$L$$