Harnessing Pairwise Ranking Prompting Through Sample-Efficient Ranking Distillation

Junru Wu[1,*], Le Yan[1], Zhen Qin[1], Honglei Zhuang[1], Paul Suganthan G. C.[2], Tianqi Liu[1], Zhe Dong[‡], Xuanhui Wang[1] and Harrie Oosterhuis[3,†]

[1]Google DeepMind, New York City/Mountain View/Seattle, NY/CA/WA, USA
[2]Google, Zurich, Switzerland
[3]Radboud University, Nijmegen, The Netherlands

Abstract

While Pairwise Ranking Prompting (PRP) with Large Language Models (LLMs) is one of the most effective zero-shot document ranking methods, it has a quadratic computational complexity with respect to the number of documents to be ranked, as it requires an enumeration over all possible document pairs. Consequently, the outstanding ranking performance of PRP has remained unreachable for most real-world ranking applications.

In this work, we propose to harness the effectiveness of PRP through pairwise distillation. Specifically, we distill a pointwise student ranker from pairwise teacher labels generated by PRP, resulting in an efficient student model that retains the performance of PRP with substantially lower computational costs. Furthermore, we find that the distillation process can be made sample-efficient: with only 2% of pairs, we are able to obtain the same performance as using all pairs for teacher labels. Thus, our novel approach provides a solution to harness the ranking performance of PRP without incurring high computational costs during both distillation and serving.

Keywords

Distillation, Learning to Rank, Large Language Models, Computational Efficiency of Training and Inference

1. Introduction

Large Language Models (LLMs) have demonstrated incredible zero-shot performance across a wide range of natural language tasks including question answering, text summarization and ranking [1, 2]. However, LLMs need to be prompted correctly to be the most effective. For document ranking, Pairwise Ranking Prompting (PRP) [3] achieves the state-of-the-art ranking performance, even with moderate-sized open-sourced LLMs. In contrast, pointwise prompting also known as Relevance Generation (RG) [4, 5] performs much more poorly and require large-sized LLMs. The crucial difference between RG and PRP is that RG asks for the relevance of an individual document, i.e., "*How relevant is document A?*", whereas PRP asks for relative relevance differences, i.e., "*Is document A more relevant than document B?*" However, while being much more effective due to its pairwise approach, PRP requires the prompting and aggregation for all document pairs to get a final ranking, resulting in a quadratic computational complexity: $O(N^2)$. Consequently, the computational costs of PRP make it infeasible for any practical ranking setting where responsiveness is important. In stark contrast, RG only requires a single prompt per document, resulting in a more practical linear complexity: $O(N)$, but also substantially lower ranking performance.

In this paper, we aim to marry the strengths of both, i.e., inherit the *effectiveness* of PRP and the *efficiency* of RG. To this end, we propose the novel *Pairwise Ranking Distillation* (PRD) which distills the ranking ability from a pairwise LLM rater to a more efficient pointwise LLM ranker, without requiring an enumeration over all document pairs. In summary, our main contributions are:

ReNeuIR 2025 (at SIGIR 2025) – 4th Workshop on Reaching Efficiency in Neural Information Retrieval, July 17, 2025, Padua, Italy
*Corresponding author.
†Work done while Harrie was at Google DeepMind.
‡Work done while Zhe was at Google.
EMAIL: junru@google.com (J. Wu); lyyanle@google.com (L. Yan); zhenqin@google.com (Z. Qin); hlz@google.com (H. Zhuang); pachristopher@google.com (P. Suganthan G. C.); tianqiliu@google.com (T. Liu); hoogendong@gmail.com (Z. Dong); xuanhui@google.com (X. Wang); harrie.oosterhuis@ru.nl (H. Oosterhuis)

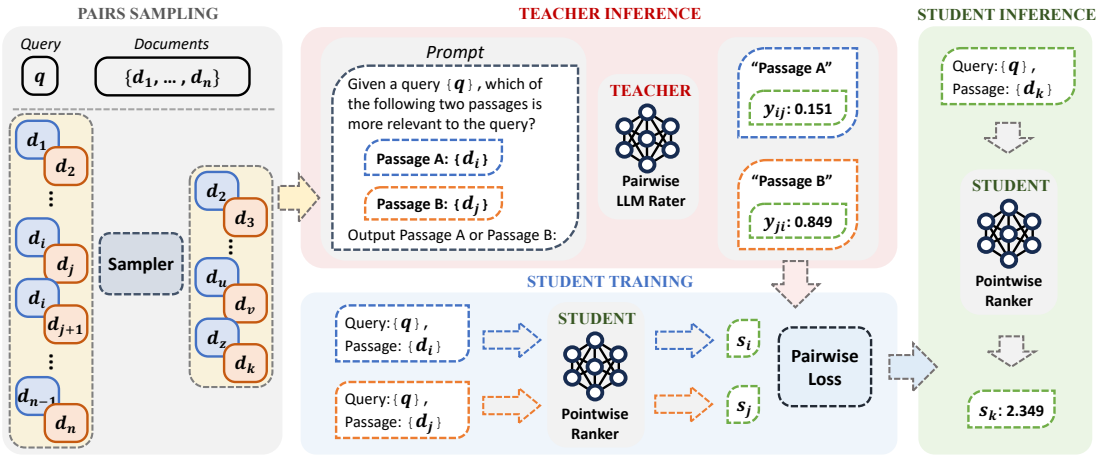


Figure 1: Visualization of the pipeline of our proposed Pairwise Ranking Distillation (PRD) method. It can be divided into four stages (1) Pairs Sampling (2) Teacher Inference (3) Student Training and (4) Student Inference.

- We propose the novel *Pairwise Ranking Distillation* (PRD) method to distill pairwise LLM ranking raters into pointwise student rankers. The student rankers can maintain the same ranking performance as PRP and significantly outperform the student rankers distilled from pointwise LLM ranking raters.
- We find that PRD is sample-efficient and can amortize the quadratic complexity of PRP during distillation. With only 2% of pairs, it achieves performance comparable to 100% of pairs.
- We design a novel ranking-aware sampling scheme that takes the order of documents in an early-stage ranking into consideration. This scheme needs less than 2% of pairs for distillation without sacrificing performance, saving more training time.

2. Related Work

Ranking with LLMs. There have been many existing efforts to use LLM as zero-shot rankers in ranking tasks, this includes pointwise [5, 6], pairwise [3] and listwise [7] approaches. By prompting LLMs differently, they are able to achieve different trade-offs between efficiency and effectiveness. Among these zero-shot LLM rankers, listwise ranker are the most efficient but also the least effective, due to the limited long-context capabilities of LLMs. Conversely, pairwise rankers (i.e., through PRP) are the most effective but least efficient due to their quadratic complexity from enumerating all pairs. On the other hand, pointwise rankers (i.e., through RG) are the most widely-used in real-world deployment due to their scalability. The pointwise rankers effectiveness and efficiency falls between those of pairwise and listwise rankers, thus a natural extension could attempt to distill the strength of pairwise PRP approach, into the more scalable and efficient pointwise ranker.

Ranking Distillation. Several existing works focus on distilling large neural rankers into more compacts models [8] through Knowledge Distillation (KD) [9]. Tang and Wang [10] use a pointwise sigmoid cross-entropy loss and only optimize on positive examples from top-k ranked samples. The later Rankdistil [11] performs listwise optimization that penalizes large scores for items ranked low by the teacher to preserve its top-k ordering.

Pairwise Distillation. Instruction Distillation [12] distills the pairwise ranking capability of LLMs into a simpler but more efficient pointwise ranking model. More recently, PAIRDISTILL [13] also aim to distill pairwise rankings, but into a dense retrieval models, which is usually a bi-encoder model. Different from our PRD approach, both Huang and Chen [13] use pairwise teacher supervision on top

of a pointwise loss, and both Huang and Chen [13] and Sun et al. [12] only use a naive random strategy for sampling pairs. Unfortunately, none of these methods have been compared with any pointwise distillation baselines, therefore, it remains unclear how much can be gained from distilling a pairwise LLM teacher over a pointwise one.

3. Problem Formulation and Background

In the relevance ranking setting, the goal is to rank a candidate set of documents, $D = \{d_1, ..., d_n\}$, for a given query. We use $Y = \{y_1, ..., y_n\}$ to denote their ground-truth labels, where y_i is the ground-truth relevance of document d_i to the query q. The optimal ranking, $R = (r_1, ..., r_n)$, orders the documents according to their ground-truth relevance in descending order, where documents with higher relevance should have smaller indices in R. The problem is that the ground-truth labels are not available during inference, and thus the ranking has to be based on other features of the documents.

Pointwise LLM Ranker. A straight-forward solution is to let an LLM directly predict the relevance labels, and subsequently, produce a ranking by sorting according to the predicted relevance. This approach of predicting the relevance per individual query-document pair is named the *pointwise* approach. A specific example is the Relevance Generation (RG) approach proposed by Liang et al. [4] and Sun et al. [5]. Here an LLM prompted with $\mathcal{I}_{\text{RG}}$="*Does the passage {document} answer the query {query}? Output Yes or No:* " for each query-document pair. Then, the probability of the LLM generating the generated tokens "*Yes*" or "*No*" are normalized into query-document relevance scores following:

$$s_{q,d}^{\text{RG}} = \frac{p(\text{“}\textit{Yes}\text{”} \mid \mathcal{I}_{RG}(q,d))}{p(\text{“}\textit{Yes}\text{”} \mid \mathcal{I}_{RG}(q,d)) + p(\text{“}\textit{No}\text{”} \mid \mathcal{I}_{RG}(q,d))}. \tag{1}$$

In practice, for instruction-tuned LLMs, the sum of the two likelihoods is often very close to one, so one can directly use the unnormalized "*Yes*" probability. The resulted ranking R is obtained by sorting according to the relevance scores.

Pairwise LLM Ranker. LLMs have been found to be better at ranking with a pairwise prompting approach [3], where the LLM is asked to judge a pair of documents and query on whether one document is more relevant to the query than the other. PRP uses the following pairwise prompt: $\mathcal{I}_{\text{PRP}}$="*Which of the following two passages is more relevant to the query {query}? Passage A: {document A}; Passage B: {document B}; Output Passage A or Passage B:* ". For each query-documents triplet (q, d_i, d_j), one of the following comparison results is obtained:

$$c_{ij} = \begin{cases} 1, & \text{if } f(\mathcal{I}_{PRP}(q, d_i, d_j)) = d_i, \\ 0, & \text{if } f(\mathcal{I}_{PRP}(q, d_i, d_j)) = d_j, \\ 0.5, & \text{otherwise.} \end{cases} \tag{2}$$

where $f(\cdot)$ denotes the LLM output given the input prompt. To obtain the final ranking, first an aggregation is performed over all pair comparisons with the following scoring scheme [3]:

$$s_{q,d}^{\text{PRP}} = \sum_{j \neq i} \left[c_{ij} + (1 - c_{ji}) \right]. \tag{3}$$

Then a ranking is created by sorting according to the scores. Because LLMs can be sensitive to the order of i and j in the prompt, comparisons in both directions are considered (both c_{ij} and c_{ji}).

4. Method: Pairwise Ranking Distillation

Figure 1 shows the full pipeline of our proposed Pairwise Ranking Distillation (PRD) method, which efficiently leverages pairwise scores of LLM teachers to distill pointwise student rankers. Our PRD

consists of four stages: (1) Pairs Sampling, (2) Teacher Inference, (3) Student Training, and (4) Student Inference. In this section, we further detail the distillation objective and the pair sampling of PRD.

4.1. Distillation Objective

Given the teacher pseudo-label pairs, y_{ij} and y_{ji}, we consider the Pairwise Logistic Ranking Loss [**?** 14], which aims to encourage models to assign higher scores to positive instances than negative instances. This can be formulated as the following loss function:

$$\mathcal{L} = \sum_{\langle i,j \rangle} \mathbf{1}_{y_{ij}<y_{ji}} \log\left(1 + \exp\left(s_i - s_j\right)\right), \tag{4}$$

where s_i and s_j are the student predictions on query-documents pairs (q, d_i) and (q, d_j), respectively. The summation is over all sampled pairs, the sampling procedure is described in Section 4.2.

4.2. Efficient Pair Sampling

The distillation objective defined in Section 4.1 involves a summation over a set of sampled query-document pairs. This set could include all possible pairs, similar to the original Pairwise Ranking Prompting [3] approach resulting in $N^2 - N$ pairs, which becomes prohibitively expensive for larger numbers of documents N. However, unlike the setting in Qin et al. [3], we have access to the initial ranking from relatively cheap (pointwise) rankers, such as a cheaper pointwise LLM ranker. To avoid the quadratic costs, we propose a number of novel efficient sampling strategies based on the initial ranking to drastically reduce the cost of invoking pairwise LLM rankers.

Random Sampling. The most straightforward method is uniform random sampling. For all the possible pairs (d_i, d_j) where $i \neq j$, we uniformly randomly sample k pairs and obtain their pairwise comparison prediction from the LLM pairwise ranker.

Reciprocal Rank Weighted (RR). To capture the most prominent ranking metrics, e.g., the Mean Reciprocal Rank [15]. Accordingly, we propose a sampling strategy that is more likely to sample pairs that would be helpful to correct top-ranked items. This strategy assumes access to an initial ranking obtained by a pointwise ranking method such as a pointwise LLM ranker, where the ranked position of each d_i is denoted by $r_i \in \{1, 2, \ldots, n\}$. For all the possible pairs (d_i, d_j) where $i \neq j$, we can assign an unnormalized weight w_{ij} as the reciprocal rank of the first document d_i: $w_{ij} = \frac{1}{r_i}$. Then a fixed number k of pairs (d_i, d_j) are sampled, without replacement, from all the possible pairs with a probability proportional to the weights w_{ij} of each pair, which gives pairs with a high-ranked document in the initial ranking a higher chance to be re-examined by the more accurate pairwise LLM ranker.

Reciprocal Rank Sum Weighted (RRSum). As a natural extension of the RR method, we propose a strategy that considers the reciprocal rank of both items by taking the average reciprocal rank. Analogous to the RR strategy, the RRSum weights for each pair (d_i, d_j) are defined as: $w_{ij} = \frac{1}{2}\left(\frac{1}{r_i} + \frac{1}{r_j}\right)$. Correspondingly, RRSum also samples k pairs, without replacement, with a probability proportional to its unnormalized weights.

Reciprocal Rank Diff Weighted (RRDiff). Another possible intuition is to sample pairs with the largest potential impact on the overall ranking quality relative to the initial ranking quality. We note that several learning-to-rank algorithms follow similar intuitions [14, 16, 17]. Accordingly, we propose RRDiff which assigns weights to document pairs based on the *absolute difference* between their reciprocal ranks. We define these weights for each pair (d_i, d_j) as: $w_{ij} = \left|\frac{1}{r_i} - \frac{1}{r_j}\right|$. Again, k pairs are sampled, without replacement, with probabilities proportional to the defined weights. Unlike RRSum, RRDiff can give pairs of documents that are both ranked higher in the initial ranking lower weights than pairs of documents where one document is ranked much higher than the other.

Table 1
Results on TREC-DL datasets with BM25 retrieved top-100 passages. Best performing system in each model variant is marked in bold. *Agg.* denotes whether to do full-pair aggregation according to Eq. 3. We used the random sampling strategy for sampling the subsets of training pairs.

LLM Rater	Model	Agg.	#Pairs	TREC-DL2019		TREC-DL2020		TREC-DL2021		TREC-DL2022		Average	
				OPA	nDCG@10	OPA	nDCG@10	OPA	nDCG@10	OPA	nDCG@10	OPA	nDCG@10
-	BM25	-	-	59.76	46.22	70.57	55.20	57.18	34.68	45.69	27.41	58.30	40.88
Teacher													
Pointwise	PaLM 2 L	-	-	83.70	74.13	81.76	69.73	89.90	81.20	82.98	69.59	84.59	73.66
Pairwise		-	-	88.33	79.05	82.83	70.13	91.53	83.41	85.88	72.06	87.14	76.16
Student													
Pointwise	Gemma1 2B	-	-	75.60	68.39	75.46	67.22	74.60	60.43	71.42	50.15	74.27	61.55
Pairwise (PRD)		Yes	100%	87.78↑**12.1%**	79.45↑**11.0%**	82.36↑**6.9%**	72.86↑**5.6%**	**88.16**↑**13.5%**	**76.33**↑**15.9%**	**82.55**↑**11.1%**	**71.65**↑**21.5%**	85.21↑**10.9%**	**75.07**↑**13.5%**
		No	100%	**88.18**↑**12.5%**	**80.11**↑**11.7%**	**83.62**↑**8.1%**	**73.58**↑**6.3%**	87.81↑**13.2%**	74.30↑**13.8%**	82.17↑**10.7%**	68.26↑**18.1%**	**85.45**↑**11.1%**	74.06↑**12.5%**
		No	2%	84.09↑**8.4%**	70.59↑**2.2%**	82.29↑**6.8%**	73.36↑**6.3%**	87.65↑**13.0%**	75.03↑**14.6%**	82.25↑**10.8%**	68.39↑**18.2%**	84.07↑**9.8%**	71.85↑**10.3%**
Pointwise	Gemma2 2B	-	-	80.83	67.37	84.40	**74.56**	86.17	74.87	77.00	61.08	82.10	69.47
Pairwise (PRD)		Yes	100%	**88.24**↑**7.4%**	79.46↑**12.1%**	83.10↓**1.3%**	73.06↓**1.5%**	88.77↑**2.6%**	**77.99**↑**3.1%**	**84.62**↑**7.6%**	**75.24**↑**14.1%**	86.18↑**4.0%**	76.44↑**6.9%**
		No	100%	88.19↑**7.3%**	**83.39**↑**16.0%**	83.48↓**0.92%**	72.59↓**1.97%**	**89.03**↑**2.8%**	77.68↑**2.8%**	84.09↑**7.1%**	73.75↑**12.6%**	**86.20**↑**4.1%**	**76.85**↑**7.4%**
		No	2%	85.06↑**4.2%**	75.16↑**7.8%**	**85.38**↑**0.9%**	74.46↓**0.1%**	87.80↑**1.6%**	75.33↑**0.46%**	84.00↑**7.0%**	69.52↑**8.4%**	85.56↑**3.4%**	73.62↑**4.1%**
Pointwise	Gemma1 7B	-	-	82.78	69.98	77.13	70.21	82.26	72.03	77.42	59.44	79.90	67.92
Pairwise (PRD)		Yes	100%	87.70↑**4.9%**	**80.54**↑**10.5%**	80.68↑**3.5%**	68.50↓**1.7%**	**83.76**↑**1.5%**	**72.37**↑**0.3%**	**81.79**↑**4.3%**	**69.26**↑**9.8%**	**83.48**↑**3.5%**	**72.67**↑**4.7%**
		No	100%	**87.85**↑**5.1%**	78.51↑**8.5%**	80.98↑**3.8%**	70.95↑**0.7%**	83.35↑**1.1%**	66.79↓**5.2%**	80.55↑**3.1%**	65.55↑**6.1%**	83.19↑**3.3%**	70.45↑**2.5%**
		No	2%	86.23↑**3.4%**	79.45↑**9.4%**	**81.30**↑**4.2%**	**71.71**↑**1.5%**	83.72↑**1.4%**	70.55↓**1.4%**	79.29↑**1.8%**	63.32↑**3.8%**	82.64↑**2.7%**	71.26↑**3.3%**

5. Experiments and Results

5.1. Datasets.

The TREC (Text REtrieval Conference) dataset refers to a collection of datasets, that offers a common ground for evaluating different information retrieval systems in IR Research [**?**]. TREC-DL (Deep Learning Track) was specifically design for evaluating the task of passage re-ranking [**? ?**]. TREC-DL 2019, 2020, 2021, 2022 subsets contains 43, 54, 53, 76 queries, respectively. For each query, we used the initial ranking of 100 passages from a IR system (i.e., BM25 score), then re-rank the passages based on their relevance of an answer to the question. We split the dataset by queries in a 7:1:2 split as training / validation / testing sets for supervised fine-tuning.

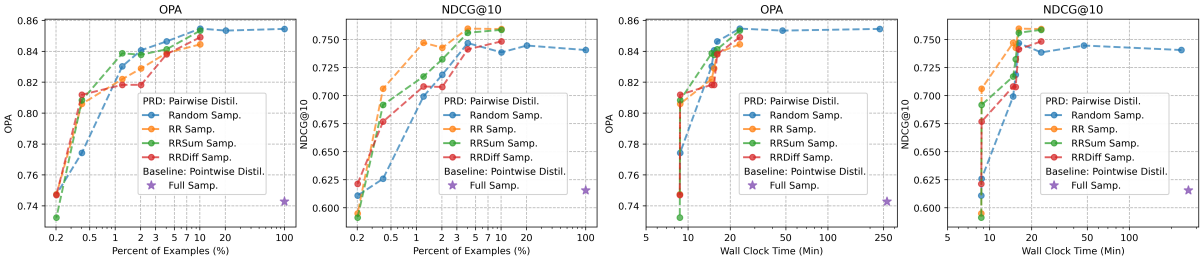
5.2. Implementation Details.

Teacher Model. We use instruction-tuned PaLM 2-L [18] as our pairwise LLM rater teacher, due to it being proven to be particular effective in the pairwise prompting setting due to its large model capacity. Similar to Pairwise Prompt Ranking [3], we used the *scoring LLMs APIs* to avoid potential issues with the generation API as it is prone to generate undesired outputs. Specifically, we used the same pairwise prompt as Qin et al. [3], and used the log probability of token "*Passage A*" and "*Passage B*" as the output of the pairwise LLM rater teacher, as discussed in detail in Section 3.

Student Model. Decoder-only architecture are widely used in modern Large Language Models (LLMs) such as GPT [19], Llama [20, 21] and Gemma [22, 23]. However, such decoder-only architecture generate tokens autoregressively, and are designed for generative tasks only, such as question answering, summarization, etc. As a result, decoder-only architectures are generally not suitable for discriminative tasks that require single or multiple scalars as output. In light of this, we convert a decoder-only model to become encoder-only following Gemma Encoder[**?**], such that the model can output a single scalar for ranking regression. We use Gemma1-2B, Gemma1-7B, and Gemma2-2B from the instruction-tuned Gemma1 [22] and Gemma2 [23] model families as our model backbones and initialization checkpoints.

5.3. Experimental Results

We compare our proposed Pairwise Ranking Distillation (PRD) with the following baselines: (a) BM25, which is an unsupervised method based on weighted term frequency; (b) Pointwise Teacher, which is a



(a) OPA over % of training examples (sampled pairs).

(b) OPA over wall-clock time.

Figure 2: Comparison of different sampling strategies on TREC-DL with Gemma1-2B as student backbone.

pointwise LLM rater as the teacher described in Section 3; (c) Pairwise Teacher, which is the pairwise LLM rater as the teacher described in Section 3; (d) Pointwise Student, which is the pointwise LLM rater as the teacher described in Section 3 to finetune an encoder-only student model; (e) Pairwise Student w/ Aggregation, which is the pairwise LLM rater as the teacher described in Section 3 to finetune a encoder-only student model, we do full-pair aggregation following Eq. 3; (f) Pairwise Student w/o Aggregation, which is our proposed Pairwise Ranking Distillation (PRD) approach. We were unable to repeat these runs due to the high costs of training each student model.

Our experiments on TREC-DL represent passage re-ranking tasks; and our metrics are the pairwise ordered pair accuracy (OPA) and the listwise NDCG@10. Overall, our PRD method achieves competitive performance; we observe the following in Table 1 and Figure 2a:

- PRD outperforms pointwise distillation in all cases by a significant margin (e.g., 3%-13% improvement in OPA and NDCG metric), regardless of the model backbone, and across all datasets
- There is no meaningful difference between full-pair aggregation according to Eq. 3 and PRD sampling all the pairs independently.
- By employing a simple yet effective random sampling scheme, PRD achieves comparable performance to full pairs with only 2% of the pairs, demonstrating its high sample-efficiency and significant saving in training wall-clock time.
- Furthermore, ranking-aware sampling techniques, in particular RR, further boosts the performance of ranking distillation when sampling less than 2% of the pairs.

6. Conclusion

In this paper, we proposed Pairwise Ranking Distillation (PRD), a method that distills the ranking ability of a complex pairwise LLM rater into a more efficient pointwise LLM ranker. Importantly, it does so without requiring an enumeration of all possible document pairs during training or inference, thus avoiding a quadratic complexity.

We found our proposed pairwise distillation scheme significantly outperforms approaches using the pointwise LLM ranking rater. Our results indicate that the distillation process can be performed sample-efficiently; by employing a simple yet effective random sampling scheme, we achieve comparable performance against full pairs, while using only 2% of pairs and still surpassing the model distilled by a pointwise LLM rater.

Moreover, we also proposed several ranking-aware sampling schemes that samples pairs base on a first-stage retrieval ranking of documents. With these schemes, we need less than 2% of the all pairs to reach optimal performance after distillation.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools for writing the text in this paper.

References

[1] OpenAI, Gpt-4 technical report, 2023. arXiv:2303.08774.

[2] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al., Gemini: a family of highly capable multimodal models, arXiv preprint arXiv:2312.11805 (2023).

[3] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, L. Yan, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang, M. Bendersky, Large language models are effective text rankers with pairwise ranking prompting, in: K. Duh, H. Gomez, S. Bethard (Eds.), Findings of the Association for Computational Linguistics: NAACL 2024, Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 1504–1518. URL: https://aclanthology.org/2024.findings-naacl.97. doi:10.18653/v1/2024.findings-naacl.97.

[4] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, et al., Holistic evaluation of language models, arXiv preprint arXiv:2211.09110 (2022).

[5] W. Sun, L. Yan, X. Ma, P. Ren, D. Yin, Z. Ren, Is ChatGPT good at search? investigating large language models as re-ranking agent, arXiv preprint arXiv:2304.09542 (2023).

[6] H. Zhuang, Z. Qin, K. Hui, J. Wu, L. Yan, X. Wang, M. Berdersky, Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels, arXiv preprint arXiv:2310.14122 (2023).

[7] R. Pradeep, S. Sharifymoghaddam, J. Lin, Rankvicuna: Zero-shot listwise document reranking with open-source large language models, arXiv preprint arXiv:2309.15088 (2023).

[8] Z. Qin, R. Jagerman, R. K. Pasumarthi, H. Zhuang, H. Zhang, A. Bai, K. Hui, L. Yan, X. Wang, Rd-suite: A benchmark for ranking distillation, Advances in Neural Information Processing Systems 36 (2023).

[9] G. Hinton, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531 (2015).

[10] J. Tang, K. Wang, Ranking distillation: Learning compact ranking models with high performance for recommender system, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 2289–2298.

[11] S. Reddi, R. K. Pasumarthi, A. Menon, A. S. Rawat, F. Yu, S. Kim, A. Veit, S. Kumar, Rankdistil: Knowledge distillation for ranking, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 2368–2376.

[12] W. Sun, Z. Chen, X. Ma, L. Yan, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Instruction distillation makes large language models efficient zero-shot rankers, arXiv preprint arXiv:2311.01555 (2023).

[13] C.-W. Huang, Y.-N. Chen, Pairdistill: Pairwise relevance distillation for dense retrieval, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024), 2024.

[14] C. J. Burges, From RankNet to LambdaRank to LambdaMART: An Overview, Technical Report Technical Report MSR-TR-2010-82, Microsoft Research, 2010.

[15] N. Craswell, Mean Reciprocal Rank, Springer US, Boston, MA, 2009, pp. 1703–1703. URL: https://doi.org/10.1007/978-0-387-39940-9_488. doi:10.1007/978-0-387-39940-9_488.

[16] C. Burges, K. Svore, P. Bennett, A. Pastusiak, Q. Wu, Learning to rank using an ensemble of lambda-gradient models, in: Proceedings of the learning to rank Challenge, 2011, pp. 25–35.

[17] X. Wang, C. Li, N. Golbandi, M. Bendersky, M. Najork, The lambdaloss framework for ranking metric optimization, in: CIKM, 2018.

[18] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, et al., Palm 2 technical report, arXiv preprint arXiv:2305.10403 (2023).

[19] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., Gpt-4 technical report, arXiv preprint arXiv:2303.08774 (2023).

[20] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al., The llama 3 herd of models, arXiv preprint arXiv:2407.21783 (2024).

[21] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., Llama 2: Open foundation and fine-tuned chat models, arXiv preprint arXiv:2307.09288 (2023).

[22] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al., Gemma: Open models based on gemini research and technology, arXiv preprint arXiv:2403.08295 (2024).

[23] G. Team, M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, et al., Gemma 2: Improving open language models at a practical size, arXiv preprint arXiv:2408.00118 (2024).