Beyond Yes and No: Improving Zero-Shot LLM Rankers via Scoring Fine-Grained Relevance Labels

Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan,
Xuanhui Wang and **Michael Bendersky**
Google Research
{hlz,zhenqin,kaihuibj,junru,lyyanle,
xuanhui,bemike}@google.com

Abstract

Zero-shot text rankers powered by recent LLMs achieve remarkable ranking performance by simply prompting. Existing prompts for pointwise LLM rankers mostly ask the model to choose from binary relevance labels like "Yes" and "No". However, the lack of intermediate relevance label options may cause the LLM to provide noisy or biased answers for documents that are partially relevant to the query. We propose to incorporate fine-grained relevance labels into the prompt for LLM rankers, enabling them to better differentiate among documents with different levels of relevance to the query and thus derive a more accurate ranking. We study two variants of the prompt template, coupled with different numbers of relevance levels. Our experiments on 8 BEIR data sets show that adding fine-grained relevance labels significantly improves the performance of LLM rankers.

1 Introduction

Large language models (LLMs) such as GPT-4 (OpenAI, 2023) and PaLM 2 (Google et al., 2023) have demonstrated impressive zero-shot performance on a variety of NLP tasks. Recently, there has been a growing interest in applying LLMs to zero-shot text ranking, with remarkably impressive results. The earliest zero-shot LLM rankers are pointwise (Liang et al., 2022; Sachan et al., 2022), which score one query and one document at each time and rank the documents based on the scores. Lately, pairwise (Qin et al., 2023) and listwise (Sun et al., 2023; Ma et al., 2023) LLM rankers also show strong performance, but they cannot scale to long lists and still largely rely on a high-quality first-stage ranking.

A typical category of pointwise LLM rankers is relevance generation (Liang et al., 2022). In this method, the LLM is prompted to answer whether a document is relevant to the query (or answers the query). Existing pointwise LLM rankers mostly ask the LLM to answer "Yes" or "No" and use the predicted likelihood of these two answers to derive the ranking score for the given query-document pair. Nevertheless, documents in many datasets are not always entirely relevant or irrelevant to the query. Some documents may not be primarily intended to answer the query, but still contain helpful information. There is no accurate mapping between these documents and the binary options.

Studies on human subjects show that using binary options sometimes lead to biased answers (Rivera-Garrido et al., 2022). Instead, providing reasonably fine-grained options can lead to more reliable results (Roitero et al., 2018; Birkett, 1986; Rivera-Garrido et al., 2022; Johnston et al., 2017). Actually, in information retrieval data sets, the annotation guidelines for human annotators often employ multiple relevance levels, like the 3-level scale used in TREC-COVID (Voorhees et al., 2021) and TREC-Robust (Voorhees, 2005), as well as the 4-level scale used in TREC-DL (Craswell et al., 2020, 2021). We believe that a zero-shot LLM ranker might share the same behavior pattern with human annotators.

Therefore, we propose to explicitly provide fine-grained relevance labels in the prompt to zero-shot LLM rankers. Instead of asking the LLM to choose between two options, we provide the LLM with fine-grained relevance labels, such as: "Highly Relevant", "Somewhat Relevant" and "Not Relevant". We then collect the LLM likelihood of all the relevance labels to derive the ranking score for each query-document pair. The intuition is that the intermediate relevance labels in the prompt will serve as a "cue" to the LLM that partially relevant documents need to be distinguished from fully relevant or fully irrelevant documents. In addition, by collecting the likelihood on more fine-grained relevance labels, we can obtain a more accurate estimate of the actual relevance, and thereby derive a better ranking. It is important to note that our focus is on developing LLM rankers, which is different from LLM assessors (Faggioli et al., 2023; Thomas et al., 2023), as our goal is only to derive a high-quality ranking with accurate top-ranked documents instead of estimating the precise (and often discrete) relevance for each individual document to sort ranking systems.

We evaluate our prompts for zero-shot LLM ranking on 8 data sets from BEIR (Thakur et al., 2021). The results show that simply adding the intermediate relevance labels allows LLM rankers to achieve substantially higher ranking performance consistently across different data sets, regardless of whether the actual ground-truth labels of the data set contain multiple graded relevance levels. An in-depth analysis shows that the new prompt enables LLM rankers to distinguish documents that are indistinguishable when there are only two options provided. We believe this discovery can benefit not only text ranking applications, but other domains such as recommendations (Fan et al., 2023; Wu et al., 2023) and user rating prediction (Kang et al., 2023).

2 Related Work

Zero-shot LLM rankers. An emerging thread of research explores how to use general-purpose LLMs for zero-shot text ranking, a shift from tuning-based learning to rank on textual and traditional tabular datasets (Nogueira et al., 2019; Han et al., 2020; Zhuang et al., 2021; Nogueira et al., 2020; Zhuang et al., 2023a; Xian et al., 2022; Liu, 2009; Qin et al., 2021).

Pointwise rankers take a single query-document pair as input and return a ranking score. The ranked list is obtained by sorting documents based on their ranking scores. The ranking score is typically calculated based on how likely the document is relevant to the query (Liang et al., 2022) or how likely the query can be generated from the document (Sachan et al., 2022). Our work is most related to this line of research. We will revisit more technical details in Section 3.

Pairwise (Qin et al., 2023) and listwise (Sun et al., 2023; Ma et al., 2023; Zhuang et al., 2023b) LLM rankers take multiple documents as input and return the ranking directly. They are usually applied iteratively on smaller sets of documents and often rely on a pointwise first-stage ranker. In this paper, we only focus on pointwise LLM rankers.

Zero-shot LLM assessors. Another related research area (Faggioli et al., 2023; Thomas et al., 2023) employs LLMs as assessors. The goal of LLM assessors is to provide a relevance label for every query-document pairs, so that the label aligns with the ground-truth relevance label, potentially created by human assessors. Existing studies (Faggioli et al., 2023; Thomas et al., 2023) also prompt LLMs with fine-grained relevance labels. LLM assessors are usually used to create an evaluation data set, which can be used to reliably evaluate different ranking models. This is different from LLM rankers, which typically only need to ensure that the relative order of the top-ranked documents are accurate. A perfect LLM assessor would also be a perfect LLM ranker, but when LLM capabilities are limited, the priorities of LLM assessor and LLM ranker development diverge.

3 LLM Rankers

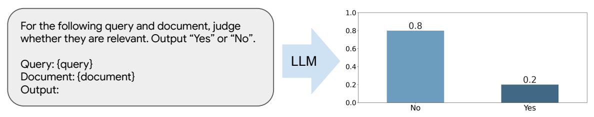
In this section, we first revisit existing pointwise LLM rankers. Then we introduce the prompting method of our LLM rankers which score fine-grained relevance labels and how we obtain the final ranking scores.

3.1 Preliminaries

Pointwise rankers. We formally describe how a pointwise ranker tackles a ranking problem. Considering a query q and a list of candidate documents $\mathbf{d} = (d_1, \ldots, d_m)$, a pointwise ranker f takes each query-document pair (q, d_i) as input and predicts a ranking score $f(q, d) \in \mathbb{R}$, which reflects the relevance of the document to the query. Once the pointwise ranker has inferred ranking scores for all documents, we can obtain a ranked list by sorting the documents based on their predicted scores.

Zero-shot LLM rankers. Existing explorations using zero-shot LLMs as pointwise rankers can be broadly divided into two categories: relevance generation (Liang et al., 2022) and query generation (Sachan et al., 2022).

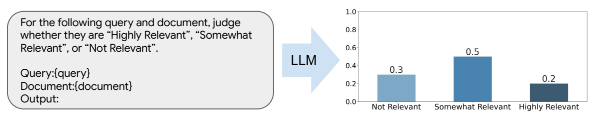
Relevance generation methods prompt the LLM with both the query q and the document d and ask whether the document is relevant to the query with “Yes” or “No” (see Figure 1(a)). To calculate the ranking score, one can use the LLM’s log-likelihood score $s_1 = \text{LLM}(\text{Yes}|q, d)$ and $s_0 = \text{LLM}(\text{No}|q, d)$, and normalize them with a



For the following query and document, judge whether they are relevant. Output "Yes" or "No".

Query: {query}
Document: {document}
Output:

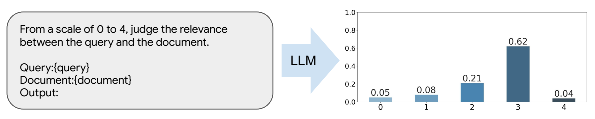
(a) Yes-No relevance generation



For the following query and document, judge whether they are "Highly Relevant", "Somewhat Relevant", or "Not Relevant".

Query:{query}
Document:{document}
Output:

(b) Fine-grained relevance label generation



From a scale of 0 to 4, judge the relevance between the query and the document.

Query:{query}
Document:{document}
Output:

(c) Rating scale relevance generation

Figure 1: Illustration of different prompting strategies for relevance generation LLM rankers.

softmax function (Nogueira et al., 2020):

$$f(q, d) = \frac{\exp(s_1)}{\exp(s_1) + \exp(s_0)}$$

Query generation methods provide the LLM with the document d as input and ask the LLM to generate a query that d answers. The ranking score is then obtained by the log-likelihood of the LLM generating the actual query q, i.e.,

$$f(q, d) = \text{LLM}(q|d)$$

We focus on relevance generation LLM rankers in this work.

3.2 Prompts

In many datasets, there exist documents that are only partially or marginally relevant to the query. These documents do not directly answer the query but may contain some relevant information. When not explicitly prompted, LLMs may struggle to decide whether to classify such documents as relevant or irrelevant.

Fine-grained relevance labels. We extend the classical relevance generation methods by introducing fine-grained relevance labels. Without loss of generality, we use a set of 3-level graded relevance labels as example: ["Not Relevant", "Somewhat Relevant", "Highly Relevant"], denoted as $[l_0, l_1, l_2]$. Then, for each query-document pair (q, d), we ask the LLM to evaluate their relevance by choosing from the given relevance labels. We can obtain the log-likelihood of the LLM generating each relevance label:

$$s_k = \text{LLM}(l_k|q, d) \tag{1}$$

This example is illustrated in Figure 1(b).

Rating scale. To avoid using relevance labels with potentially ambiguous order, we can also employ a rating scale. For example, we can prompt the LLM to rate the relevance between the query q and the document d on a scale from 0 to 4. We can then use the LLM to obtain the log-likelihood $[s_0, \ldots, s_4]$ of generating each relevance scale value $[l_0, \ldots, l_4]$, which are "0" to "4" respectively. This method allows us to try arbitrarily fine-grained relevance levels in the prompt. Figure 1(c) illustrates an example of this prompt.

3.3 Ranking Scores

Once we obtain the log-likelihood of each relevance labels, we can derive the ranking scores.

Expected relevance values (ER). The most straightforward way is to calculate the expected relevance value. To do this, we first derive the marginal probability of generating each relevance label given all the candidate relevance labels by:

$$p_k = \frac{\exp(s_k)}{\sum_{k'} \exp(s_{k'})} \tag{2}$$

Then, we can assign a series of relevance values $[y_0, y_1, y_2]$ to all the relevance labels $[l_0, l_1, l_2]$, where $y_k \in \mathbb{R}$. The relevance value should reflect the relevance degree expressed by the textual relevance label. We can then calculate the ranking score as the expected relevance value by:

$$f(q, d) = \sum p_k \cdot y_k \tag{3}$$

The relevance values y_k can be provided by users or even tuned based on a training data set. In our experiments, we find that with relevance labels starting from the least relevant to the most relevant, naïvely assigning $y_k = k$ can already provide great performance. Hence, we simply use $y_k = k$.

Peak relevance likelihood (PR). Alternatively, since LLM rankers are typically evaluated by ranking metrics which heavily focus on the accuracy of top-ranked items instead of the entire ranked list, we can further simplify the ranking score derivation by only using the log-likelihood of the relevance

Table 1: Relevance labels used in RG-kL. The relevance label with the maximum relevance value is bolded.

Method	Relevance Labels
RG-2L	"Not Relevant", **"Relevant"**
RG-3L	"Not Relevant", "Somewhat Relevant", **"Highly Relevant"**
RG-4L	"Not Relevant", "Somewhat Relevant", "Highly Relevant", **"Perfectly Relevant"**

label with the highest relevance value. For example, "Highly Relevant" is the relevance label with the highest relevance value among "Not Relevant", "Somewhat Relevant" and "Highly Relevant". We still prompt the LLM with all three relevance labels as options, but only use the log-likelihood of "High Relevant" as the ranking score.

More formally, let l_{k^*} denote the relevance label expressing the highest relevance label. We can simply rank the documents by:

$$f(q, d) = s_{k^*} \tag{4}$$

Note that s_{k^*} is the log-likelihood directly obtained from the LLM$(l_{k^*}|q, d)$, instead of the marginal probability derived from Equation (3). Hence, it is not necessary to score any other relevance labels using the LLM and could potentially save some decoding cost when using this strategy to derive the ranking score. While this method is shown less effective on smaller models (Nogueira et al., 2020), it works well empirically with larger models in our experiments.

4 Experiment Setup

Data set. We conduct experiments on 8 chosen data sets (Sun et al., 2023) from BEIR (Thakur et al., 2021): Covid, Touche, DBPedia, SciFact, Signal, News, Robust04, and NFCorpus. Notice that our method is applicable regardless of whether the data set is actually labeled with corresponding graded relevance, since the final output of our method are just real-number ranking scores.

We use BM25 (Lin et al., 2021) to retrieve the top-100 documents for each data set, and then rank the retrieved documents using LLMs with our proposed methods. We use FLAN PaLM2 S (Google et al., 2023) as the LLM in our experiments.

The ranking performance is measured by NDCG@10 (Järvelin and Kekäläinen, 2002).

Compared methods. We compared the following prompting strategies:

1. Query Generation (QG). Ranking documents based on the LLM likelihood of generating the query given the document (Sachan et al., 2022).

2. Binary Relevance Generation (RG-YN). Prompting the LLM with a query-document pair and using the likelihood of "Yes/No" to calculate the ranking score (Liang et al., 2022).

3. k-Level Relevance Generation (RG-kL). Prompting the LLM to choose from k relevance labels for each query-document pair. The relevance labels used are listed in Table 1.

4. Rating Scale 0-to-k Relevance Generation (RG-S$(0, k)$). Prompting the LLM to rate the relevance for each query-document pair using a scale from 0 to k. Notice that for RG-S$(0, k)$, the LLM needs to score the log-likelihood for $(k + 1)$ possible outputs.

The exact prompts can be found in Appendix F.

By default, the ranking scores of our proposed methods are derived using the expected relevance values as shown in Equation (3). When needed, the method name is appended with the suffix "-ER". We also conduct experiments to compare methods with ranking scores derived using peak relevance likelihood according to Equation (4), indicated by suffix "-PR".

5 Results

Overall performance. Table 2 summarizes the overall comparison results. We also plot how the performance changes with regard to k for the rating scale prompting method RG-S$(0, k)$ in Figure 2.

It can be seen that when the LLM is prompted with only 2 relevance labels (RG-YN, RG-2L), the average performance is lower. However, when the LLM is prompted with more fine-grained relevance labels, the performance can be substantially improved. RG-3L on average achieves +2% improvement in NDCG@10 compared with RG-2L and RG-YN. RG-S$(0, 4)$ which uses the rating scale 0 to 4 in the prompt also achieves similar improvement. Note that even on data sets with binary ground-truth labels (e.g., SciFact), using fine-grained relevance labels still achieves substantial improvement. This suggests that the improvement is not merely a result of matching the actual ground-truth relevance levels of the data set. Rather, the

Table 2: Overall ranking performances measured by NDCG@10 on BEIR data sets. The best performances are bolded. Average results that are significantly (paired t-test, p<0.05) better than RG-2L are marked with *.

Method	Covid	Touche	DBPedia	SciFact	Signal	News	Robust04	NFCorpus	Average
QG	0.7357	0.2408	0.3773	0.7495	0.2872	0.4156	0.4651	0.3673	0.4548
RG-YN	0.7897	0.2427	0.3696	0.6958	0.3196	0.4588	0.5656	0.3743	0.4770
RG-2L	0.7949	0.2411	0.3590	0.7290	0.2996	0.4623	0.5636	0.3814	0.4789
RG-3L	**0.8065**	0.2650	0.4013	0.7671	0.3142	**0.4890**	0.5660	0.3849	0.4992*
RG-4L	0.8063	0.2388	0.4033	**0.7766**	0.3184	0.4884	0.5635	0.3801	0.4969*
RG-S$(0, 2)$	0.7760	0.2695	0.3709	0.6921	0.3034	0.4677	0.5557	0.3787	0.4768
RG-S$(0, 4)$	0.8048	**0.2757**	**0.4190**	0.7521	**0.3301**	0.4790	**0.5668**	**0.3901**	**0.5022***

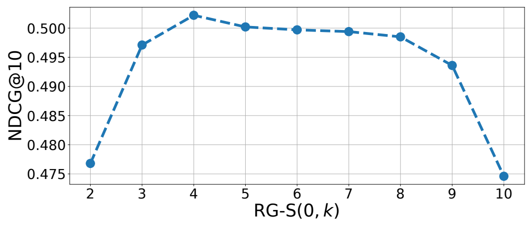


Figure 2: Comparing average NDCG@10 on 8 BEIR data sets with different number of relevance scales for the rating scale relevance generation method.

fine-grained relevance labels in the LLM prompts help it to develop a more nuanced understanding of relevance.

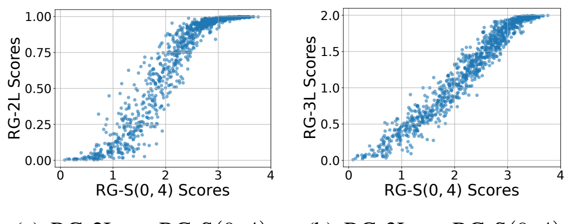
However, the exact number of fine-grained relevance labels needed to achieve the performance improvement varies across different prompts. For example, simply using 3-level textual relevance labels is sufficient to achieve average NDCG@10 close to 0.50; but using rating scale from 0 to 2, which also corresponds to 3 relevance levels, can only obtain NDCG@10 lower than 0.48. Figure 2 shows that for rating scale relevance generation RG-S$(0, k)$, the NDCG@10 only gets close to 0.50 with more than about 4 relevance levels.

On the other hand, further adding more relevance levels does not always improve the performance. For example, RG-4L performance seems to be on par with RG-3L. In Figure 2, the performance from RG-S$(0, 4)$ and RG-S$(0, 8)$ also remain similar, and the performance of RG-S$(0, 9)$ and RG-S$(0, 10)$ is even worse than RG-S$(0, 4)$.

Ranking score derivation. We also compare the two alternative strategies to derive the ranking scores from LLM likelihood scores. The results are shown in Table 3. Generally, the expected relevance values derived from the marginal probability (Equation (3)) deliver better ranking scores overall.

Table 3: Comparing different strategies to derive the ranking score. Measured by average NDCG@10 on BEIR data sets.

Prompts	Ranking Score	
	ER	PR
RG-2L	0.4789	0.4726
RG-3L	0.4992	0.5005
RG-4L	0.4969	0.4934
RG-S$(0, 2)$	0.4768	0.4659
RG-S$(0, 4)$	0.5022	0.4988



(a) RG-2L vs. RG-S$(0, 4)$ (b) RG-3L vs. RG-S$(0, 4)$

Figure 3: Comparing ranking score distribution of different methods on the Covid data set.

However, the ranking scores derived from peak relevance likelihood (Equation (4)) achieve very close performance to expected relevance values in RG-kL prompts where textual fine-grained relevance labels are used. When downstream applications of the LLM ranker are sensitive to decoding cost, the peak relevance likelihood strategy can provide a more efficient alternative.

Score distribution. We also compare the score distribution of different methods. Figure 3 shows the scattered plot of ranking scores derived from two methods for a random sample of query-document pairs in the Covid data set.

We observe that RG-2L's ranking scores are mostly positively correlated with RG-S$(0, 4)$'s

(Figure 3(a)). However, RG-2L struggles to distinguish query-document pairs with higher (> 3.0) ranking scores from RG-S$(0, 4)$ and scores them almost equally with scores close to 1.0. This suggests that providing more fine-grained relevance labels helps the LLM differentiate better among some query-document pairs, particularly with the top-ranked documents. When we compare the ranking scores from RG-3L where more than 2 relevance levels are used (Figure 3(b)), there is almost no such "plateau". The performance of RG-3L and RG-S$(0, 4)$ are also very close.

6 Conclusion

In this work, we explore the use of more fine-grained relevance labels in the prompt for pointwise zero-shot LLM rankers instead of the binary labels used in existing works. We propose to either provide intermediate relevance labels such as "Somewhat Relevant" as additional choices for the LLM or ask the LLM to rate the relevance between query-document pairs using a rating scale. Then we aggregate the likelihood of different relevance levels into ranking scores to obtain the ranked list. Our experiments on BEIR data sets demonstrate that prompting with fine-grained relevance labels can consistently improve the ranking performance across different data sets, as it enables the model to better differentiate query-document pairs potentially ranked at the top.

We believe our discovery can be further extended to applications beyond information retrieval. For example, the same method can be applied for recommendation (Fan et al., 2023; Wu et al., 2023), where the LLM is asked to rate how likely a user would buy an item.

7 Limitations

In this work, we assume that the predicted likelihood for any generated text can be accessed. However, we are aware that this might not always be true for many commercial LLMs where users can only call with specific APIs.

Another limitation is that our experiments are conducted only using one LLM, which is FLAN PaLM2 S. While we believe the results can be generalize to other LLMs, we do not have the resource to verify this.

References

Nicholas J Birkett. 1986. Selecting the number of response categories for a likert-type scale. In *Proceedings of the American statistical association*, volume 1, pages 488–492.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the TREC 2020 deep learning track. *arXiv preprint arXiv:2102.07662*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Guglielmo Faggioli, Laura Dietz, Charles LA Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, et al. 2023. Perspectives on large language models for relevance judgment. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 39–50.

Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender systems in the era of large language models (llms). *arXiv preprint arXiv:2307.02046*.

Google, Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. PaLM 2 technical report.

Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-rank with BERT in TF-Ranking. *arXiv preprint arXiv:2004.08476*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Robert J Johnston, Kevin J Boyle, Wiktor Adamowicz, Jeff Bennett, Roy Brouwer, Trudy Ann Cameron, W Michael Hanemann, Nick Hanley, Mandy Ryan, Riccardo Scarpa, et al. 2017. Contemporary guidance for stated preference studies. *Journal of the Association of Environmental and Resource Economists*, 4(2):319–405.

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Tie-Yan Liu. 2009. *Learning to Rank for Information Retrieval*. Now Publishers Inc.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 708–718.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424*.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, et al. 2023. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2021. Are neural rankers still outperformed by gradient boosted decision trees? In *International Conference on Learning Representations*.

Noelia Rivera-Garrido, MP Ramos-Sosa, Michela Accerenzi, and Pablo Brañas-Garza. 2022. Continuous and binary sets of responses differ in the field. *Scientific Reports*, 12(1):14376.

Kevin Roitero, Eddy Maddalena, Gianluca Demartini, and Stefano Mizzaro. 2018. On fine-grained relevance scales. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 675–684.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. *arXiv preprint arXiv:2204.07496*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agent. *arXiv preprint arXiv:2304.09542*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2023. Large language models can accurately predict searcher preferences. *arXiv preprint arXiv:2309.10621*.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, pages 1–12. ACM New York, NY, USA.

Ellen M Voorhees. 2005. The trec robust retrieval track. In *ACM SIGIR Forum*, volume 39, pages 11–20. ACM New York, NY, USA.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860*.

Ruicheng Xian, Honglei Zhuang, Zhen Qin, Hamed Zamani, Jing Lu, Ji Ma, Kai Hui, Han Zhao, Xuanhui Wang, and Michael Bendersky. 2022. Learning list-level domain-invariant representations for ranking. *arXiv preprint arXiv:2212.10764*.

Honglei Zhuang, Zhen Qin, Shuguang Han, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2021. Ensemble distillation for BERT-based ranking models. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 131–136.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023a. RankT5: Fine-tuning T5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313.

Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2023b. A setwise approach for effective and highly efficient zero-shot ranking with large language models. *arXiv preprint arXiv:2310.09497*.

A Alternative Relevance Levels

We replace the relevance levels with other phrases to examine how the performance changes. For RG-2L, we replace “Not Relevant” with “Irrelevant”; for RG-3L, we replace “Somewhat Relevant” with “Partially Relevant”.

The results are shown in Table 4. Regardless of using different textual representations of relevance labels, RG-3L consistently outperforms RG-2L. This suggests that the discovery in this paper is generalizable to different choices of textual relevance labels. Another observation is that RG-2L performance varies slightly more than RG-3L performance. This might indicate that RG-3L is more robust to different wording of relevance labels.

Table 4: Comparing ranking performance with different textual relevance levels. Measured by average NDCG@10 on BEIR data sets.

Method	Relevance Levels	Average
RG-2L	“Irrelevant”, “Relevant”	0.4717
	“Not Relevant”, “Relevant”	0.4789
RG-3L	“Not Relevant”, “Partially Relevant”, “Highly Relevant”	0.4975
	“Not Relevant”, “Somewhat Relevant”, “Highly Relevant”	0.4992

We also experiment with different rating scale formulation. Instead of prompting the LLM to rate the relevance from 0 to k, we also try to ask the LLM to rate the relevance from 1 to k, denoted as RG-S$(1, k)$. We plot the average NDCG@10 performance in Figure 4.

The performance of both methods do not differ much when k is larger than 4. But not providing the “0” option substantially hurt the performance when k is lower than or equal to 3. This might also suggest that using the rating scale from 0 to k is slightly more robust.

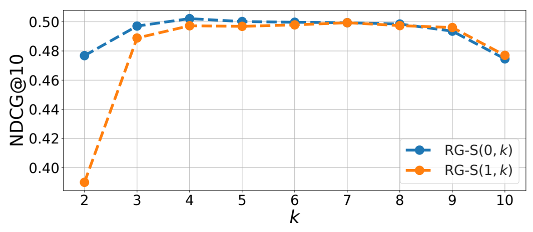


Figure 4: Comparing rating scale relevance generation with different prompts.

B In-Depth Score Distribution

We plot the in-depth score distribution of our methods. Specifically, we group the query-document pairs in Covid data set by different ground-truth relevance and plot the distribution of the marginal probability p_k for each prompted relevance label l_k respectively. Figure 5 and 6 shows the results on Covid data set when we use RG-S$(0, 4)$ and RG-4L respectively. The ground-truth relevance of Covid data set is 0, 1 or 2.

In Figure 5, We observe that the distributions of marginal probability p_k of relevance label "0", "1" and "2" shift down towards 0 as the ground-truth relevance increases. Meanwhile, the distributions of p_k across relevance label "3" and "4" shift up towards 1. In Figure 6, we found a similar trend where the distributions of marginal probability p_k of "Not Relevant" and "Somewhat Relevant" shift down towards 0 as the ground-truth relevance increases, while the distributions of p_k across "Highly Relevant" and "Perfectly Relevant" shift up towards 1. This reveals how our expected relevance values (ER) methods works in practice, and also given us hints on how peak relevance likelihood (PR) alone works based on the distribution shift of the peak relevance label.

C Varying Assigned Relevance Values

We also investigate how the user provided relevance values y_k's make a difference to the ranking performance. We use RG-3L as the example. We fix $y_0 = 0$ for "Not Relevant" and $y_2 = 2$ for "Highly Relevant", but vary the relevance value y_1 for "Somewhat Relevant" between y_0 and y_2. We evaluate the average NDCG@10 on the 8 BEIR data sets and presents the results in Table 5.

As y_1 varies, the average NDCG@10 does not change substantially when y_1 decreases. Even when $y_1 = y_0$, the NDCG@10 performance remains high. This is expected as NDCG@10 metric only focuses on the top-ranked items. Hence changing the relevance values of intermediate relevance labels may not change the order of top-ranked items a lot. This is also similar to using the peak relevance likelihood method.

In contrast, when $y_1 = y_2$, the performance drops significantly to about the same level as RG-2L. This might indirectly explain why RG-2L performance is worse than RG-3L, as it might not be able to distinguish partially relevant and highly relevant documents.

Table 5: Comparing ranking performance with different relevance values y_k's. Measured by average NDCG@10 on BEIR data sets.

Method	$[y_0, y_1, y_2]$	Average
RG-3L	$[0.00, 0.00, 2.00]$	0.5000
RG-3L	$[0.00, 0.50, 2.00]$	0.5000
RG-3L	$[0.00, 1.00, 2.00]$	0.4992
RG-3L	$[0.00, 1.50, 2.00]$	0.4990
RG-3L	$[0.00, 2.00, 2.00]$	0.4779

Table 6: Comparing ranking performance instruction and in-context learning. Measured by average NDCG@10 on BEIR data sets.

Method	Average
RG-2L	0.4789
+ Instructions	0.4914
+ Instructions + 4-shot ICL	0.4914
RG-3L	0.4992
+ Instructions	0.5034
+ Instructions + 4-shot ICL	0.5046

D Instructions and In-Context Learning

We also try adding instructions and few-shot exemplars into the prompt. For instructions, we directly add the definition of the relevance labels into the prompt. The relevance label definitions are directly copied from TREC-DL 2020 (Craswell et al., 2021). For RG-2L instructions we use the "Irrelevant" and "Relevant" labels; for RG-3L instructions we use the "Irrelevant", "Relevant" and "Highly Relevant" labels. We also change the relevance labels accordingly to align with the instructions.

In addition to instructions, we also try to include few-shot exemplars to leverage the model's in-context learning capabilities. We include 4-shot exemplars, which are randomly sampled from TREC-DL 2020 data sets. We sampled 2 "Irrelevant", 1 "Relevant" and 1 "Perfectly Relevant" query-document pairs. To align with the instructions, for RG-2L we label both "Relevant" and "Perfectly Relevant" exemplar query-document pairs as "Relevant"; for RG-3L we label the "Perfectly Relevant" pair as "Highly Relevant".

The results are shown in Table 6. Adding instructions improves both RG-2L and RG-3L, while RG-3L still remains +1.2% better than RG-2L. Further adding exemplars on top of the instructions does not improve much, possibly due to the distribution discrepancy between TREC-DL and BEIR.

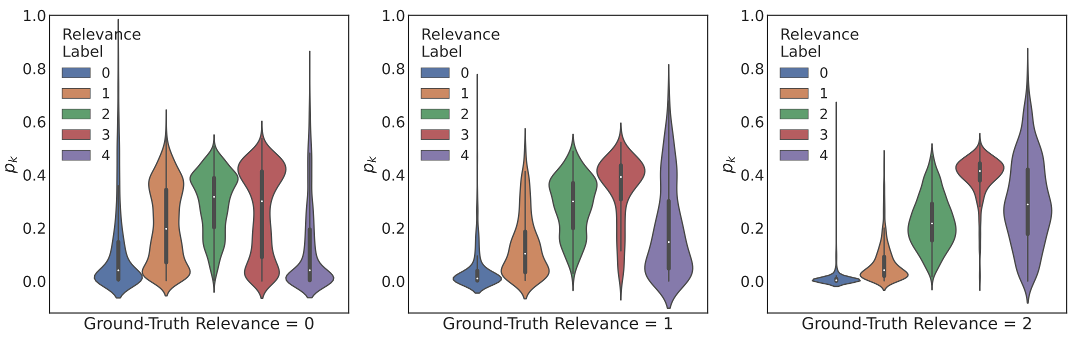


Figure 5: Distribution of marginal probability p_k of each relevance label in RG-S$(0, 4)$ for query-document pairs with different ground-truth labels on Covid data set

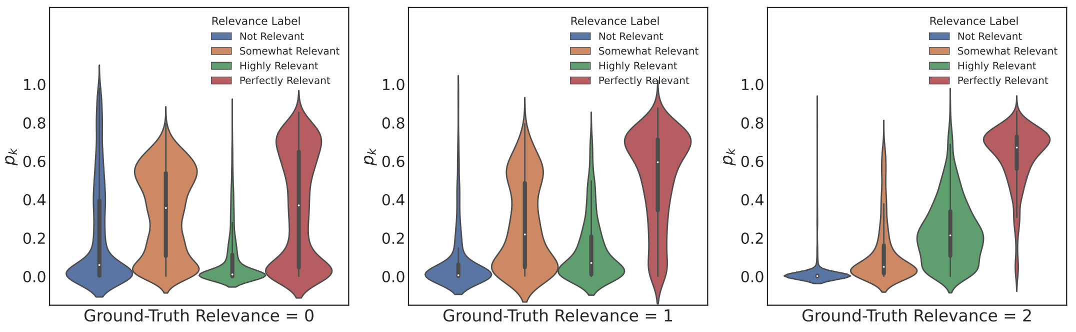


Figure 6: Distribution of marginal probability p_k of each relevance label in RG-4L for query-document pairs with different ground-truth labels on Covid data set

Table 7: Overall ranking performances measured by NDCG@10 on BEIR data sets.

Method	Model	Covid	Touche	DBPedia	SciFact	Signal	News	Robust04	NFCorpus	Average
BM25	N/A	0.5947	0.4422	0.3180	0.6789	0.3305	0.3952	0.4070	0.3075	0.4342
QG	FLAN PaLM2 S	0.7357	0.2408	0.3773	0.7495	0.2872	0.4156	0.4651	0.3673	0.4548
RG-YN	FLAN PaLM2 S	0.7897	0.2427	0.3696	0.6958	0.3196	0.4588	0.5656	0.3743	0.4770
RG-2L-ER	FLAN PaLM2 S	0.7949	0.2411	0.3590	0.7290	0.2996	0.4623	0.5636	0.3814	0.4789
RG-2L-PR	FLAN PaLM2 S	0.7874	0.2482	0.3435	0.7230	0.2819	0.4619	0.5647	0.3706	0.4726
RG-3L-ER	FLAN PaLM2 S	0.8065	0.2650	0.4013	0.7671	0.3142	0.4890	0.5660	0.3849	0.4992
RG-3L-PR	FLAN PaLM2 S	0.8065	0.2634	0.4032	0.7745	0.3202	0.4816	0.5681	0.3860	0.5005
RG-4L-ER	FLAN PaLM2 S	0.8063	0.2388	0.4033	0.7766	0.3184	0.4884	0.5635	0.3801	0.4969
RG-4L-PR	FLAN PaLM2 S	0.8076	0.2354	0.4050	0.7772	0.3121	0.4712	0.5561	0.3824	0.4934
RG-S$(0, 2)$-ER	FLAN PaLM2 S	0.7760	0.2695	0.3709	0.6921	0.3034	0.4677	0.5557	0.3787	0.4768
RG-S$(0, 2)$-PR	FLAN PaLM2 S	0.7821	0.2735	0.3469	0.6954	0.2597	0.4540	0.5409	0.3752	0.4659
RG-S$(0, 4)$-ER	FLAN PaLM2 S	0.8048	0.2757	0.4190	0.7521	0.3301	0.4790	0.5668	0.3901	0.5022
RG-S$(0, 4)$-PR	FLAN PaLM2 S	0.8036	0.2785	0.4221	0.7625	0.3168	0.4623	0.5559	0.3886	0.4988
monoT5	Fine-tuned T5 XL	0.8071	0.3241	0.4445	0.7657	0.3255	0.4849	0.5671	0.3897	0.5136
RankT5	Fine-tuned T5 XL	0.8200	0.3762	0.4419	0.7686	0.3180	0.4815	0.5276	0.3860	0.5150
RankGPT	GPT-3.5 Turbo	0.7667	0.3618	0.4447	0.7043	0.3212	0.4885	0.5062	0.3562	0.4937
PRP	UL2	0.7945	0.3789	0.4647	0.7333	0.3520	0.4911	0.5343	N/A	N/A

E More Comparison Results

We also include a more thorough comparison with other methods including:

- BM25. The base retriever performance.
- monoT5 (Nogueira et al., 2020). A T5 XL model fine-tuned on MS MARCO data set for text ranking task and applied directly on the BEIR data sets.
- RankT5 (Zhuang et al., 2023a). An encoder-only model initialized with T5 XL but fine-tuned on MS MARCO data set using listwise softmax cross-entropy ranking loss and applied directly on the BEIR data sets.

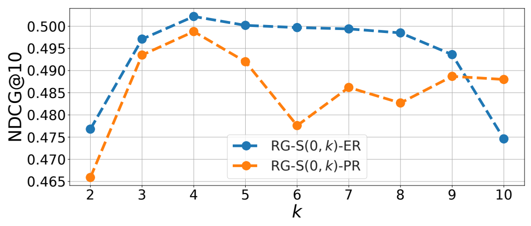


Figure 7: Comparing rating scale relevance generation with different strategies to derive ranking scores.

- Pairwise Ranking Prompts (PRP) (Qin et al., 2023). A zero-shot pairwise LLM ranker which takes a query and two documents as input, and outputs which one is more relevant to the query. We include the best results of PRP which uses UL2 as the LLM and a sliding window strategy.

- RankGPT (Sun et al., 2023). A zero-shot listwise LLM ranker which takes a query and a list of documents as input, and outputs an ordered list of documents based on their relevance. The method is used jointly with a sliding window strategy. We do not include the GPT-4 reranking number as it involves a second-stage ranking.

We also include the detailed results of our proposed methods with two strategies of derive ranking scores. Table 7 illustrates the results. Figure 7 also plots the performance of rating scale methods ranking score derivation methods.

It is not surprising that our methods perform slightly worse than monoT5 or RankT5 as they are fine-tuned for the text ranking task on MS MARCO data set. However, it is encouraging to see our prompting method substantially shrinks the gap between zero-shot LLM rankers and RankT5.

Our methods can also perform slightly better than single-stage RankGPT. When compared with PRP, our methods can achieve better or close performance to 5 out of 7 overlapping data sets except Touche and DBPedia. However, note that the LLM used in these experiments are different, so the difference might also be explained by the model difference.

F Prompts

In this section, we provide the prompts we used for each method:

F.1 Query Generation (QG)

We use the following prompt for our QG experiments. We find this prompt performs better empirically for zero-shot QG LLM rankers than the prompt used in existing works (Sachan et al., 2022).

```
I will check whether what you said could answer my question.

You said: {document}
I googled: {query}
```

F.2 Binary Relevance Generation (RG-YN)

We use the following prompt for our RG-YN experiments. We find this prompt performs better empirically than the prompt used originally by Liang et al. (2022), Sun et al. (2023) and Qin et al. (2023).

```
For the following query and document, judge whether they are relevant. Output “Yes” or “No”.

Query: {query}
Document: {document}
Output:
```

F.3 2-Level Relevance Generation (RG-2L)

```
For the following query and document, judge whether they are “Relevant”, or “Not Relevant”.

Query: {query}
Document: {document}
Output:
```

F.4 3-Level Relevance Generation (RG-3L)

```
For the following query and document, judge whether they are “Highly Relevant”, “Somewhat
Relevant”, or “Not Relevant”.

Query: {query}
Document: {document}
Output:
```

F.5 4-Level Relevance Generation (RG-4L)

```
For the following query and document, judge whether they are “Perfectly Relevant”, “Highly
Relevant”, “Somewhat Relevant”, or “Not Relevant”.

Query: {query}
Document: {document}
Output:
```

F.6 Rating Scale Relevance Generation (RG-S$(0, k)$)

```
From a scale of 0 to {k}, judge the relevance between the query and the document.

Query: {query}
Document: {document}
Output:
```